Matematica delle Reti Neurali

Volume V

Silvano Salvador

Modelli Grafici Probabilistici · Geometria dell'Informazione · Neural ODE e Fisica
Inferenza Causale · Geometria Profonda · Processi Gaussiani · Boosting · Online Learning

Sezioni §276–341 — Fondamenti avanzati dell'apprendimento automatico moderno

PGMFisher InformationNatural Gradient Neural ODEHMC & NUTSPINN Do-CalculusHyperbolic NNGaussian Processes Bayesian OptAdaBoostSVM Duale FTRLOptimal Transport

Bayesian Networks: Struttura, d-Separazione e Inferenza Esatta

Distribuzioni di probabilità congiunte fattorizzate tramite DAG — dalla combinatoria esponenziale alla rappresentazione compatta.

276.1 · Il Problema della Distribuzione Congiunta

Una distribuzione di probabilità congiunta su n variabili binarie richiede in linea di principio $2^n - 1$ parametri per essere specificata completamente: un numero che per $n = 100$ supera il numero di atomi nell'universo osservabile. Il miracolo pratico che rende possibile il ragionamento probabilistico su sistemi complessi è la struttura di indipendenza condizionale: nella maggior parte dei domini reali, ogni variabile dipende direttamente soltanto da un piccolo sottoinsieme delle altre. Le **Bayesian Networks** — introdotte da Judea Pearl nel suo seminale libro *Probabilistic Reasoning in Intelligent Systems* (1988) e poi formalizzate in tutta la loro potenza matematica nel decennio successivo — codificano questa struttura di indipendenza tramite un grafo aciclico diretto (DAG) e fattorizzano la congiunta come prodotto di distribuzioni condizionali locali.

Un DAG $\mathcal{G} = (V, E)$ con n nodi, uno per ogni variabile aleatoria, definisce la fattorizzazione:

$$P(X_1, \ldots, X_n) = \prod_{i=1}^{n} P(X_i \mid \mathrm{pa}_{\mathcal{G}}(X_i))$$

dove $\mathrm{pa}_{\mathcal{G}}(X_i)$ denota l'insieme dei genitori di X_i nel grafo. Questa fattorizzazione non è un'assunzione arbitraria ma una conseguenza della **condizione di Markov locale**: ogni variabile è condizionalmente indipendente dai propri non-discendenti dati i propri genitori. Se ogni nodo ha al più k genitori, il numero di parametri si riduce da $O(2^n)$ a $O(n \cdot 2^k)$, un risparmio

che per k piccolo è esponenziale. La struttura del DAG non determina soltanto la fattorizzazione numerica: essa codifica un modello qualitativo del dominio che può essere costruito con l'aiuto di esperti di dominio prima ancora di raccogliere un singolo dato — una caratteristica che distingue le reti bayesiane dai metodi puramente data-driven.

276.2 · d-Separazione: la Geometria dell'Indipendenza

La domanda centrale nell'inferenza su una rete bayesiana è: dati due insiemi di variabili A e B e un insieme di evidenze C, è vero che $A \perp\!\!\!\perp B \mid C$? La risposta è data da un criterio puramente grafico — la **d-separazione** — che opera sull'algoritmo di percorso non orientato nel DAG senza richiedere la conoscenza numerica dei parametri. Un percorso non orientato tra due nodi è *bloccato* da un insieme C se esiste almeno un nodo intermedio Z sul percorso che soddisfa uno dei seguenti pattern: nelle strutture a catena $A \to Z \to B$ o a fork $A \leftarrow Z \to B$, il percorso è bloccato se $Z \in C$ (l'osservazione di Z "blocca" il flusso di informazione); nella struttura a collider $A \to Z \leftarrow B$, il percorso è bloccato se $Z \notin C$ e nessun discendente di Z appartiene a C (l'osservazione del collider, paradossalmente, *apre* il flusso). Due insiemi A e B sono d-separati da C se tutti i percorsi non orientati tra loro sono bloccati, e in tal caso $A \perp\!\!\!\perp B \mid C$ in ogni distribuzione compatibile con il DAG.

Il pattern del collider merita attenzione speciale poiché è radicalmente controintuitivo: due cause indipendenti di un effetto comune diventano dipendenti quando si condiziona sull'effetto. L'esempio classico è noto come *paradosso di Berkson*: tra i pazienti ricoverati in ospedale (condizionamento sull'effetto "ricovero"), due cause indipendenti di ricovero risultano negativamente correlate. Lo stesso fenomeno — chiamato anche *explaining away* o *inter-causal reasoning* — è alla base di molti errori di ragionamento statistico e di bias sistematici nei dataset osservazionali costruiti per selezione.

276.3 · Eliminazione delle Variabili e Complessità dell'Inferenza

L'inferenza esatta in una rete bayesiana — calcolare la distribuzione posteriore di una variabile query X_q date le evidenze $\mathbf{X}_e = \mathbf{x}_e$ — richiede la marginalizzazione della distribuzione congiunta rispetto a tutte le variabili latenti. L'algoritmo di **eliminazione delle variabili** esegue questa marginalizzazione in modo efficiente sfruttando la proprietà distributiva della somma rispetto al prodotto, unendo i fattori che coinvolgono la variabile da eliminare prima della sommatoria. La sequenza con cui le variabili vengono eliminate determina la *treewidth* del grafo, che governa la complessità computazionale: trovare l'ordine di eliminazione ottimale è NP-hard, ma euristiche greedy come *min-fill* (minimizzare il numero di archi aggiunti al grafo morale) e *min-degree* (scegliere la variabile con meno vicini nel grafo morale corrente) danno risultati eccellenti nella

pratica. Per reti con struttura ad albero o polibero, l'eliminazione richiede tempo lineare nel numero di nodi — base dell'algoritmo belief propagation trattato nella §279.

Markov Random Fields: Energia, Potenziali e Teorema di Hammersley-Clifford

Modelli grafici non orientati per dipendenze simmetriche — dalla fisica statistica al riconoscimento d'immagine.

277.1 · Dipendenze Simmetriche e la Necessità dei Grafi Non Orientati

Le Bayesian Networks descrivono relazioni asimmetriche con una direzione causale esplicita. Esistono tuttavia molti domini in cui le dipendenze tra variabili sono intrinsecamente simmetriche: i pixel vicini in un'immagine si influenzano reciprocamente senza che uno "causi" l'altro; gli spin vicini in un materiale ferromagnetico si allineano a vicenda; le parole adiacenti in un testo vincolano la distribuzione di probabilità l'una dell'altra in modo bidirezionale. Per questi contesti i **Markov Random Fields (MRF)**, detti anche *Markov Undirected Models* o semplicemente modelli a grafo non orientato, offrono la rappresentazione naturale. Un MRF è un grafo non orientato $\mathcal{G} = (V, E)$ in cui ogni variabile è condizionalmente indipendente da tutte le variabili non-adiacenti dati i propri vicini immediati — la **global Markov property** rispetto al grafo.

Il teorema fondamentale dei MRF è il **Teorema di Hammersley-Clifford** (1971), che stabilisce l'equivalenza tra la global Markov property e la fattorizzazione in potenziali di clique: per qualunque distribuzione strettamente positiva $P(\mathbf{x}) > 0$ che soddisfi la global Markov property rispetto a $\mathcal{G}$, esiste una fattorizzazione in *clique potentials*:

$$P(\mathbf{x}) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \psi_C(\mathbf{x}_C), \qquad Z = \sum_{\mathbf{x}} \prod_{C} \psi_C(\mathbf{x}_C)$$

dove il prodotto è sulle clique massimali $\mathcal{C}$ del grafo, $\psi_C \geq 0$ è una funzione non negativa di compatibilità della configurazione locale, e Z è la **funzione di partizione**. Quest'ultima è il collo di bottiglia computazionale per l'inferenza nei MRF: calcolarla esattamente richiede di sommare su tutte le $|\mathcal{X}|^{|V|}$ configurazioni, un problema #P-hard in generale. Scrivendo ogni potenziale come $\psi_C(\mathbf{x}_C) = e^{-E_C(\mathbf{x}_C)}$ si ottiene la **distribuzione di Gibbs** $P(\mathbf{x}) = e^{-E(\mathbf{x})}/Z$, dove $E(\mathbf{x}) = \sum_C E_C(\mathbf{x}_C)$ è l'energia totale del sistema. Il collegamento con la meccanica statistica non è superficiale: Ising (1925) aveva introdotto esattamente questo formalismo per modellare

la transizione di fase nei materiali ferromagnetici, e la comunità del machine learning ha poi trasferito decenni di intuizioni e tecniche dalla fisica teorica all'elaborazione dell'informazione.

277.2 · Il Modello di Ising e le Boltzmann Machine

Il **modello di Ising** su griglia bidimensionale con variabili di spin $x_i \in \{-1, +1\}$ ha energia $E(\mathbf{x}) = -J \sum_{(i,j)\in E} x_i x_j - h \sum_i x_i$, dove il primo termine premia la coerenza locale tra spin vicini (accoppiamento ferromagnetico per $J > 0$) e il secondo l'allineamento col campo esterno h. Questo modello esibisce una transizione di fase del secondo ordine a una temperatura critica T_c: al di sotto di T_c il sistema si magnetizza spontaneamente anche in assenza di campo esterno. Hinton e Sejnowski (1985) hanno generalizzato il modello di Ising al caso delle **Boltzmann Machine**, con energie $E(\mathbf{v}, \mathbf{h}) = -\mathbf{b}^T\mathbf{v} - \mathbf{c}^T\mathbf{h} - \mathbf{v}^T W h$ su unità visibili $\mathbf{v}$ e nascoste $\mathbf{h}$. Il training richiede di valutare $\mathbb{E}_{P(\mathbf{v},\mathbf{h})}[v_i h_j]$, un'aspettativa sotto la distribuzione del modello che richiede il calcolo di Z — intrattabile per reti dense. Questa difficoltà ha motivato le **Restricted Boltzmann Machine (RBM)**, dove l'assenza di connessioni intra-layer (né tra visibili né tra nascoste) rende le unità di ogni layer condizionalmente indipendenti dato l'altro layer, permettendo il Gibbs sampling a blocchi alternato in modo esatto.

Factor Graphs: Rappresentazione Unificante per BN e MRF

Il formalismo bipartito che unifica Bayesian Networks e MRF in un unico framework algoritmico per il belief propagation.

278.1 · Il Grafo Bipartito Fattore-Variabile

Le Bayesian Networks e i Markov Random Fields sono spesso trattati come paradigmi separati con algoritmi distinti — eliminazione delle variabili per BN, loopy BP per MRF — ma Kschischang, Frey e Loeliger (2001) hanno dimostrato che entrambi sono istanze di un framework unificante: i **Factor Graphs**. Un factor graph è un grafo bipartito con due tipi di nodi: *nodi variabile* (cerchi) e *nodi fattore* (quadrati). Un arco connette il fattore f_a alla variabile x_i se e solo se x_i è un argomento di f_a. La distribuzione si scrive:

$$P(\mathbf{x}) = \frac{1}{Z} \prod_{a \in F} f_a(\mathbf{x}_{\partial a})$$

dove ∂a è il neighborhoodhood del fattore a nel grafo bipartito. La conversione da BN a factor graph è immediata: per ogni variabile X_i si crea un fattore $f_i(\mathbf{x}_{\{i\}\cup\text{pa}(i)}) = P(x_i \mid \text{pa}(x_i))$. La conversione da MRF avviene creando un fattore $f_C = \psi_C$ per ogni potenziale di clique. Il vantaggio del formalismo unificante è che un unico algoritmo — il Sum-Product — si applica a entrambi, e riduce a tutti gli algoritmi di inferenza esatta noti per casi speciali: forward-backward per HMM, algoritmo di Viterbi, filtro di Kalman per modelli lineari-gaussiani, BCJR per canali a memoria, turbo decoding. Questa convergenza ha avuto un impatto profondo nella teoria dei codici a correzione di errore, dove la comprensione del belief propagation come sum-product su factor graph ha portato alla progettazione di codici LDPC vicini al limite di Shannon.

278.2 · Eliminazione e Treewidth come Proprietà del Factor Graph

La complessità dell'inferenza esatta su un factor graph è determinata dalla sua *treewidth*: per grafi ad albero (nessun ciclo) il sum-product richiede tempo lineare; per grafi con cicli corti e densi può essere esponenziale. La *junction tree algorithm* converte un factor graph con cicli in un albero di cluster (il *junction tree*) dove ogni nodo è un cluster di variabili, e applica il sum-product sull'albero di cluster. La dimensione dei cluster è determinata dalla treewidth del grafo originale: per grafi planari la treewidth è $O(\sqrt{n})$, per grafi di griglia 2D è $O(\sqrt{n})$, per grafi di griglia 3D è $O(n^{2/3})$ — crescita che spiega perché l'inferenza esatta sulla griglia 3D è intrattabile già per griglie di dimensione moderata e motiva gli approcci approssimati della §279.

Belief Propagation: Sum-Product, Max-Product e Loopy BP

Passaggio locale di messaggi per l'inferenza distribuita — esatto su alberi, empiricamente potente su grafi con cicli.

279.1 · L'Algoritmo Sum-Product

Il **Sum-Product Algorithm**, detto anche **Belief Propagation (BP)**, calcola le marginali di una distribuzione fattorizzata tramite scambio iterativo di messaggi tra nodi variabile e nodi fattore nel factor graph. Su un factor graph aciclico (albero), due passaggi di messaggi — dalle foglie alla radice e di ritorno — producono le marginali esatte di tutte le variabili simultaneamente, con costo lineare nel numero di archi. I messaggi hanno forme precise: il messaggio dal nodo variabile x_i al fattore f_a è il prodotto di tutti i messaggi ricevuti dagli altri

fattori vicini, escluso f_a:

$$\mu_{x_i \to f_a}(x_i) = \prod_{b \in \partial x_i \setminus a} \mu_{f_b \to x_i}(x_i)$$

Il messaggio dal fattore f_a alla variabile x_i è la somma (marginalizzazione) del prodotto del fattore e di tutti i messaggi ricevuti dalle altre variabili vicine, escluso x_i:

$$\mu_{f_a \to x_i}(x_i) = \sum_{\mathbf{x}_{\partial a \setminus i}} f_a(\mathbf{x}_{\partial a}) \prod_{j \in \partial a \setminus i} \mu_{x_j \to f_a}(x_j)$$

La marginale di ogni variabile è proporzionale al prodotto di tutti i messaggi ricevuti dai fattori vicini: $P(x_i) \propto \prod_{a \in \partial x_i} \mu_{f_a \to x_i}(x_i)$. L'intuizione geometrica è che ogni messaggio riassume tutta l'informazione rilevante proveniente dal "ramo" del grafo a cui il fattore o la variabile appartiene — una forma di propagazione locale dell'informazione che diventa globalmente esatta sulla struttura ad albero. Il costo computazionale per messaggio è $O(|\mathcal{X}|^{|\partial a|})$, dove $|\mathcal{X}|$ è la cardinalità dello spazio di ogni variabile e $|\partial a|$ il grado del fattore.

279.2 · Max-Product e Decodifica MAP

Sostituendo ogni somma con un massimo nel calcolo dei messaggi si ottiene l'algoritmo **Max-Product** (o Max-Sum nel dominio logaritmico), che calcola la configurazione a massima probabilità posteriore $\mathbf{x}^* = \arg\max_{\mathbf{x}} P(\mathbf{x} \mid \mathbf{x}_e)$. Il Max-Product sul factor graph di una HMM coincide esattamente con l'algoritmo di Viterbi (1967), usato in milioni di applicazioni di speech recognition, allineamento di sequenze biologiche e decodifica di canali. La phase di backtracking per recuperare la configurazione ottimale (non solo il suo valore) richiede di memorizzare per ogni variabile il valore che ha realizzato il massimo durante il passaggio forward — equivalente alla tabella di memoizzazione del Viterbi.

279.3 · Loopy Belief Propagation e la Bethe Free Energy

Il **Loopy Belief Propagation** applica il Sum-Product a factor graph con cicli ignorando la struttura non-albero: i messaggi vengono inizializzati e poi aggiornati in modo iterativo (schedule parallelo o sequenziale) fino alla convergenza. Non vi è garanzia né di convergenza né di correttezza sui grafi con cicli, ma empiricamente l'algoritmo produce soluzioni eccellenti in molti contesti di grande importanza pratica. I codici LDPC (Low-Density Parity-Check) per la correzione di errore, i factor graph con cicli lunghi e rari, e i grafi di Ising al di sopra

della temperatura critica sono tutti casi dove il loopy BP converge a soluzioni quasi-ottimali. Yedidia, Freeman e Weiss (2001) hanno dimostrato che i punti fissi del loopy BP corrispondono ai punti stazionari della **Bethe Free Energy**, un'approssimazione variazionale dell'energia libera di Helmholtz che usa le "beliefs" locali invece della distribuzione globale: $F_{\text{Bethe}} = \sum_a E_a(b_a) - \sum_i (d_i - 1) H(b_i)$, dove d_i è il grado del nodo variabile i. Questo legame variazionale ha aperto la via alle *Generalized Belief Propagation*, dove le beliefs sono definite su cluster di variabili anziché su singole variabili, migliorando sistematicamente le approssimazioni a spese di un maggiore costo computazionale.

Mean Field Variational Inference: ELBO, CAVI e Posteriors Fattorizzate

L'inferenza bayesiana come ottimizzazione — la famiglia fattorizzata, il coordinate ascent e i limiti dell'approssimazione.

280.1 · Dall'Integrazione all'Ottimizzazione

L'ostacolo fondamentale dell'inferenza bayesiana esatta è il calcolo dell'evidenza marginale $P(\mathbf{x}) = \int P(\mathbf{x} \mid \mathbf{z}) P(\mathbf{z})\, d\mathbf{z}$: un integrale che per modelli con likelihood non coniugata alla prior — praticamente tutti i modelli neurali — non ha forma analitica. L'**inferenza variazionale** converte il problema di integrazione in un problema di ottimizzazione: si definisce una famiglia di distribuzioni trattabili $\mathcal{Q}$ e si cerca il membro $q^* \in \mathcal{Q}$ più vicino alla vera posterior $P(\mathbf{z} \mid \mathbf{x})$, dove la vicinanza è misurata dalla divergenza KL $\mathrm{KL}(q \| P(\cdot \mid \mathbf{x}))$. Poiché $\mathrm{KL}(q\|p) = \mathbb{E}_q[\log q(\mathbf{z})] - \mathbb{E}_q[\log P(\mathbf{x}, \mathbf{z})] + \log P(\mathbf{x})$, minimizzare la KL è equivalente a massimizzare l'**Evidence Lower BOund (ELBO)**:

$$\mathcal{L}(q) = \mathbb{E}_{q(\mathbf{z})}[\log P(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z})] = \log P(\mathbf{x}) - \mathrm{KL}(q(\mathbf{z}) \| P(\mathbf{z} \mid \mathbf{x}))$$

L'ELBO è un lower bound al log-evidence $\log P(\mathbf{x})$ — da cui il nome — poiché la KL è non negativa. Massimizzare l'ELBO significa contemporaneamente stringere il bound e avvicinare q alla posterior: quando $\mathcal{L}(q) = \log P(\mathbf{x})$, si ha $q = P(\cdot \mid \mathbf{x})$ e la KL è nulla. Il connettore con l'algoritmo EM è diretto: il passo E massimizza $\mathcal{L}(q)$ rispetto a q con parametri fissi (proiezione m-flat nella geometria dell'informazione), e il passo M massimizza $\mathcal{L}(q)$ rispetto ai parametri con q fissa (proiezione e-flat).

280.2 · La Famiglia Mean Field e l'Aggiornamento Ottimale

Il **Mean Field** assume la fattorizzazione completa $q(\mathbf{z}) = \prod_{i=1}^{m} q_i(z_i)$, eliminando ogni correlazione tra le variabili nella distribuzione approssimante. È l'assunzione più forte possibile di indipendenza nella familia variazionale. Derivando la condizione di stazionarietà dell'ELBO rispetto a ogni fattore q_j mantenendo fissi tutti gli altri, si ottiene la forma ottimale:

$$\log q_j^*(z_j) = \mathbb{E}_{q_{-j}}[\log P(\mathbf{x}, \mathbf{z})] + \text{const}$$

dove $\mathbb{E}_{q_{-j}}$ denota l'aspettativa rispetto a tutte le variabili tranne z_j. L'algoritmo di **Coordinate Ascent Variational Inference (CAVI)** aggiorna ciclicamente ogni q_j alla sua forma ottimale dato lo stato corrente degli altri fattori, con convergenza garantita al massimo locale dell'ELBO per la proprietà di ascesa monotona delle coordinate. Per modelli nella famiglia esponenziale coniugata, questi aggiornamenti hanno forma chiusa: i parametri naturali della distribuzione ottimale q_j^* sono la somma dei parametri naturali del prior e dell'aspettativa delle statistiche sufficienti delle variabili con cui z_j interagisce. Il nome "mean field" deriva dalla fisica statistica: ogni variabile vede le altre come se fossero fissate ai propri valori medi, analogamente all'approssimazione di campo medio nei sistemi magnetici dove ogni spin è soggetto a un campo effettivo generato dalla magnetizzazione media dei vicini.

280.3 · Limitazioni: Sottostima della Varianza e Posteriors Multimodali

Il mean field produce sistematicamente due tipi di distorsione. La prima è la sottostima della varianza delle singole variabili: poiché la fattorizzazione ignora le correlazioni, l'incertezza marginale di ogni z_j non può beneficiare dell'informazione portata dalle correlazioni, e la distribuzione risultante è più concentrata della vera posterior. La seconda e più grave distorsione è l'incapacità di modellare posteriors multimodali: la minimizzazione di $\mathrm{KL}(q\|p)$ penalizza fortemente $q(z) > 0$ nelle regioni dove $p(z \mid \mathbf{x})$ è piccolo, portando la distribuzione ottimale a concentrarsi su uno solo dei modi della posterior e ignorare gli altri. Questo comportamento è noto come *mode-seeking* o *zero-forcing*: l'approssimazione è "sicura" nel senso di evitare regioni improbabili, ma non è "conservativa" nel senso di coprire tutta la distribuzione. La direzione opposta $\mathrm{KL}(p\|q)$, usata nell'EP e in alcune varianti di inferenza variazionale, produce invece soluzioni *mass-covering* che tendono a coprire tutti i modi a spese di una massa non nulla nelle valli tra i modi.

Expectation Propagation: Moment Matching e Cavity Distribution

Inferenza approssimata con KL nella direzione inversa — approssimazione fattore per fattore, gaussiane assunte, convergenza.

281.1 · L'Idea del Matching dei Momenti

L'**Expectation Propagation (EP)**, sviluppato da Thomas Minka nella sua tesi di dottorato al MIT (2001, lavoro basato sull'Assumed Density Filtering di Lauritzen), combina la struttura del belief propagation con l'approssimazione variazionale in un modo che aggira le limitazioni del mean field rispetto alle posteriors multimodali. Invece di approssimare la posterior globale con una distribuzione fattorizzata, EP approssima ogni *fattore locale* $f_a(\mathbf{x})$ della distribuzione con un fattore semplice $\tilde{f}_a(\mathbf{x})$ nella famiglia esponenziale, e usa un procedimento iterativo di aggiornamento per rendere queste approssimazioni locali globalmente consistenti. L'aggiornamento di ogni fattore approssimante avviene tramite **moment matching**, che equivale a minimizzare $\mathrm{KL}(p_{\text{tilt}} \| \tilde{f}_a)$ (direzione inversa rispetto al mean field) dove $p_{\text{tilt}} \propto q^{\setminus a} \cdot f_a$ è la distribuzione *tilted* che sostituisce l'approssimazione corrente con il fattore vero.

Il procedimento EP si svolge in tre passi per ogni fattore f_a. Si calcola prima la **cavity distribution** $q^{\setminus a}(\mathbf{z}) \propto q(\mathbf{z})/\tilde{f}_a(\mathbf{z})$, ottenuta dividendo la distribuzione corrente per il contributo dell'approssimazione del fattore a — questa operazione rimuove l'influenza di $\tilde{f}_a$ dalla distribuzione globale. Si calcola poi la distribuzione **tilted** $\hat{p}_a(\mathbf{z}) \propto q^{\setminus a}(\mathbf{z}) \cdot f_a(\mathbf{z})$, che sostituisce l'approssimazione con il fattore vero reintegrandolo nel contesto della cavity. Infine si effettua il moment matching: si trova la nuova distribuzione q_{new} nella famiglia esponenziale che ha gli stessi momenti sufficienti di $\hat{p}_a$, e si aggiorna $\tilde{f}_a = q_{\text{new}}/q^{\setminus a}$. Per fattori gaussiani e approssimanti gaussiani, tutti questi passi hanno forma chiusa e riducono a operazioni di propagazione di media e varianza — da cui la connessione con il filtro di Kalman esteso e il filtro di Kalman unscented, che sono casi speciali di EP per sistemi di state-space.

281.2 · EP Gaussiano e Applicazioni

Quando la famiglia approssimante è gaussiana e la distribuzione target non è gaussiana a causa di likelihood "difficili" (probit, logistica, Student-t, distribuzioni con supporto limitato), EP produce approssimazioni notevolmente più accurate del mean field nella direzione della kurtosi e delle correlazioni tra variabili. Per la regressione probit bayesiana, EP produce stime dell'incertezza quasi indistinguibili dal campionamento MCMC con un costo computazionale molto inferiore. Per i Processi Gaussiani con classification likelihood (§317), EP è il metodo di riferimento per la stima delle marginali. Un limite pratico di EP è la sua possibile mancanza

di convergenza: la sequenza degli aggiornamenti non è monotonamente crescente rispetto a nessuna funzione obiettivo nota in generale, e su certi modelli EP oscilla o diverge, richiedendo strategie di damping (aggiornamenti parziali $q \leftarrow q^{1-\alpha} \cdot q_{\text{new}}^{\alpha}$ con $\alpha < 1$).

Black-Box Variational Inference e il Reparameterization Trick

ELBO con gradienti Monte Carlo — REINFORCE, reparameterization trick, VAE come inferenza variazionale amortizzata.

282.1 · Il Collo di Bottiglia della Coniugazione

La forza del CAVI — aggiornamenti in forma chiusa — è anche la sua debolezza: richiede che ogni variabile z_j abbia una distribuzione condizionale $P(z_j \mid z_{-j}, \mathbf{x})$ che appartenga alla famiglia esponenziale coniugata, limitando di fatto l'applicabilità ai modelli iperparametrici classici (mixture models, LDA, modelli di stato). I modelli neurali, con likelihood definite da reti profonde e trasformazioni non lineari, non rientrano in questa classe. La **Black-Box Variational Inference (BBVI)** di Ranganath, Gerrish e Blei (AISTATS 2014) generalizza l'inferenza variazionale a qualunque modello di cui si possa valutare il log-joint, usando stime Monte Carlo del gradiente dell'ELBO.

Il gradiente di $\mathcal{L}(q_\phi) = \mathbb{E}_{q_\phi(\mathbf{z})}[\log P(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z})]$ rispetto ai parametri ϕ della famiglia variazionale si calcola con il **log-derivative trick** (detto anche REINFORCE, Williams 1992):

$$\nabla_\phi \mathcal{L} = \mathbb{E}_{q_\phi(\mathbf{z})}[(\log P(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z}))\, \nabla_\phi \log q_\phi(\mathbf{z})]$$

stimabile con campioni $\mathbf{z}^{(s)} \sim q_\phi$. Il problema centrale è la varianza altissima di questo stimatore: il fattore moltiplicativo $\log P(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z})$ varia enormemente tra campioni, rendendo l'ottimizzazione instabile. Tecniche di riduzione della varianza come le *baseline* sottrattive — si sottrae una costante b dalla parentesi, tipicamente la media empirica del fattore, senza introdurre bias — sono essenziali per la convergenza pratica.

282.2 · Reparameterization Trick e la Connessione con le VAE

Il **reparameterization trick** (Kingma e Welling 2013, Rezende, Mohamed e Wierstra 2014) è la soluzione elegante e definitiva al problema della varianza elevata per distribuzioni differen-

ziabili: invece di campionare $\mathbf{z} \sim q_\phi(\mathbf{z} \mid \mathbf{x})$ direttamente, si esprime il campione come una trasformazione deterministica differenziabile di rumore esterno $\boldsymbol{\epsilon}$ e dei parametri ϕ:

$$\mathbf{z} = g_\phi(\mathbf{x}, \boldsymbol{\epsilon}), \quad \boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon}), \qquad \nabla_\phi \mathcal{L} \approx \frac{1}{S} \sum_{s=1}^{S} \nabla_\phi \left[\log P(\mathbf{x}, g_\phi(\boldsymbol{\epsilon}^{(s)})) - \log q_\phi(g_\phi(\boldsymbol{\epsilon}^{(s)})) \right]$$

Per una gaussiana diagonale $q_\phi(\mathbf{z} \mid \mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}_\phi(\mathbf{x}), \mathrm{diag}(\boldsymbol{\sigma}^2_\phi(\mathbf{x})))$, si scrive $\mathbf{z} = \boldsymbol{\mu}_\phi(\mathbf{x}) + \boldsymbol{\sigma}_\phi(\mathbf{x}) \odot \boldsymbol{\epsilon}$ con $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Il gradiente scorre ora attraverso g_ϕ in modo differenziabile, riducendo la varianza di ordini di grandezza rispetto a REINFORCE. Il **Variational Autoencoder (VAE)** (Kingma e Welling 2013) è l'istanza più famosa di BBVI con reparameterization: l'encoder $q_\phi(\mathbf{z} \mid \mathbf{x})$ è una rete neurale che produce media e varianza per la distribuzione variazionale, il campionamento avviene tramite reparameterization, e il decoder $p_\theta(\mathbf{x} \mid \mathbf{z})$ massimizza la ricostruzione. L'ELBO del VAE si scompone naturalmente in un termine di ricostruzione $\mathbb{E}_{q_\phi}[\log p_\theta(\mathbf{x} \mid \mathbf{z})]$ e un termine di regolarizzazione $-\mathrm{KL}(q_\phi(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z}))$ che per prior gaussiana ha forma chiusa: questo secondo termine spinge la distribuzione delle rappresentazioni latenti verso la prior, garantendo che lo spazio latente sia continuo e interpolabile.

MCMC: Metropolis-Hastings, Gibbs Sampling e Diagnostica della Convergenza

Campionamento esatto dalla posterior — detailed balance, mixing, catene di Markov ergodiche.

283.1 · Il Principio del Markov Chain Monte Carlo

L'inferenza variazionale approssima la posterior con una distribuzione trattabile; il **Markov Chain Monte Carlo (MCMC)** la approssima con un campione proveniente — asintoticamente — dalla distribuzione esatta. Il principio è di costruire una catena di Markov $\theta_1, \theta_2, \ldots$ su Θ che sia ergodica con distribuzione stazionaria uguale alla posterior $P(\theta \mid \mathbf{x})$. Per la legge dei grandi numeri delle catene di Markov, per qualunque funzione f, la media temporale $\frac{1}{T}\sum_{t=1}^{T} f(\theta_t)$ converge quasi certamente a $\mathbb{E}_{P(\theta|\mathbf{x})}[f(\theta)]$. La varianza della stima è $\mathrm{Var}[\hat{f}] = \frac{1}{T}\mathrm{Var}[f(\theta)](1 + 2\tau_{\mathrm{int}})$, dove τ_{int} è il tempo di mixing integrato — una misura della correlazione tra campioni successivi che dipende dalla geometria della posterior e dall'efficacia del kernel di transizione. L'MCMC non introduce bias strutturale: al contrario dell'inferenza variazionale, l'unica fonte di errore è statistica e diminuisce con $O(1/\sqrt{T})$, a patto di superare la fase di burn-in (rilassamento verso la stazionaria) e di avere un mixing sufficiente.

L'algoritmo **Metropolis-Hastings (MH)** (Metropolis et al. 1953, Hastings 1970) è il kernel di transizione fondamentale: data una distribuzione di proposta $q(\theta' \mid \theta)$, si propone $\theta' \sim q(\cdot \mid \theta_t)$ e si accetta con probabilità $\alpha = \min(1, \frac{P(\theta'|\mathbf{x})\, q(\theta_t|\theta')}{P(\theta_t|\mathbf{x})\, q(\theta'|\theta_t)})$. La **detailed balance** $P(\theta)T(\theta' \mid \theta) = P(\theta')T(\theta \mid \theta')$ garantisce che $P(\cdot \mid \mathbf{x})$ sia la distribuzione stazionaria. Crucialmente, il rapporto $P(\theta' \mid \mathbf{x})/P(\theta_t \mid \mathbf{x}) = P(\mathbf{x} \mid \theta')P(\theta')/[P(\mathbf{x} \mid \theta_t)P(\theta_t)]$ cancella l'evidenza $P(\mathbf{x})$ intrattabile: MH richiede soltanto di valutare il log-joint $\log P(\mathbf{x}, \theta)$ a meno di una costante di normalizzazione, rendendolo universalmente applicabile.

283.2 · Gibbs Sampling e il Mixing Lento

Il **Gibbs Sampling** (Geman e Geman, 1984, originariamente per la segmentazione MAP di immagini con MRF) è un caso speciale di MH con tasso di accettazione $\alpha = 1$, ottenuto campionando ogni variabile dalla propria distribuzione condizionale completa: $\theta_i^{(t+1)} \sim P(\theta_i \mid \theta_{-i}^{(t)}, \mathbf{x})$. Quando queste condizionali sono in forma chiusa — come nei modelli coniugati della famiglia esponenziale — Gibbs è computazionalmente efficiente e facile da implementare. Il suo principale difetto è il *mixing lento* in presenza di forti correlazioni tra variabili: la catena si muove lungo una variabile alla volta e non può "saltare" attraverso valli di bassa probabilità che separano modi diversi della posterior. Per una distribuzione gaussiana bivariata con correlazione ρ vicina a 1, il tempo di mixing di Gibbs è $\tau \sim 1/(1-\rho^2)$ che diverge per $\rho \to 1$, mentre HMC (§284) scala molto meglio. Tecniche come il *block Gibbs sampling* (campionare blocchi di variabili correlate congiuntamente) e gli *auxiliary variable methods* (come l'algoritmo di Swendsen-Wang per il modello di Ising) mitigano il mixing lento a spese di un maggiore costo computazionale per iterazione.

Hamiltonian Monte Carlo: Leapfrog, NUTS e la Geometria del Campionamento

La dinamica hamiltoniana come meccanismo di proposta — moti globali nello spazio delle fasi, HMC vs RWMH in alta dimensione.

284.1 · Il Problema del Random Walk ad Alta Dimensione

Il Metropolis-Hastings con proposta random-walk $\theta' = \theta + \epsilon$ (RWMH) soffre di un problema fondamentale in alta dimensione: il volume della sfera di raggio r in d dimensioni è concentrato sulla superficie, e la frazione di volume accettata dalla proposta decresce esponenzialmente con d. Per ottenere un tasso di accettazione del 23% (ottimale per RWMH gaussiano con posterior gaussiana), la dimensione del passo deve scalare come $\sigma_{\text{prop}} \sim d^{-1/2}$, producendo passe sempre

più piccoli man mano che la dimensionalità cresce e tempi di mixing $O(d)$. L'**Hamiltonian Monte Carlo (HMC)** sfugge a questa trappola simulando la dinamica hamiltoniana in uno spazio delle fasi esteso $(\theta, \mathbf{p})$ dove θ sono i parametri e $\mathbf{p}$ sono momenti ausiliari. L'energia è $\mathcal{H}(\theta, \mathbf{p}) = U(\theta) + K(\mathbf{p})$ con $U(\theta) = -\log P(\theta \mid \mathbf{x})$ (energia potenziale) e $K(\mathbf{p}) = \frac{1}{2}\mathbf{p}^T M^{-1}\mathbf{p}$ (energia cinetica con matrice di massa M). La distribuzione congiunta $\pi(\theta, \mathbf{p}) \propto e^{-\mathcal{H}(\theta,\mathbf{p})}$ ha marginalizzata su $\mathbf{p}$ uguale a $P(\theta \mid \mathbf{x})$ — la distribuzione target.

284.2 · L'Integratore Leapfrog e la Proprietà Simplettica

Le equazioni hamiltoniane $\dot{\theta} = \partial\mathcal{H}/\partial\mathbf{p} = M^{-1}\mathbf{p}$ e $\dot{\mathbf{p}} = -\partial\mathcal{H}/\partial\theta = \nabla_\theta \log P(\theta \mid \mathbf{x})$ devono essere integrate numericamente. L'integrazione deve essere *reversibile nel tempo* (per soddisfare detailed balance) e *simplettica* (volume-preserving nello spazio delle fasi, per garantire che la distribuzione target sia stazionaria). L'**integratore leapfrog** (Störmer-Verlet) soddisfa entrambe le proprietà:

$$\mathbf{p}_{t+\varepsilon/2} = \mathbf{p}_t + \frac{\varepsilon}{2}\nabla_\theta \log P(\theta_t \mid \mathbf{x})$$

$$\theta_{t+\varepsilon} = \theta_t + \varepsilon M^{-1}\mathbf{p}_{t+\varepsilon/2}$$

$$\mathbf{p}_{t+\varepsilon} = \mathbf{p}_{t+\varepsilon/2} + \frac{\varepsilon}{2}\nabla_\theta \log P(\theta_{t+\varepsilon} \mid \mathbf{x})$$

Il leapfrog è esatto fino a $O(\varepsilon^3)$ per passo e conserva l'energia hamiltoniana fino a $O(\varepsilon^2)$ per traiettorie di L passi. Invertendo il segno del momento $\mathbf{p} \to -\mathbf{p}$ dopo L passi si ottiene la traiettoria inversa — questa reversibilità garantisce la detailed balance anche per proposte a grande distanza. Il passo di accettazione/rifiuto MH correttivo per l'errore di discretizzazione porta a tassi di accettazione tipicamente superiori al 65-90% con passo ε ben calibrato, producendo salti molto più grandi nello spazio dei parametri rispetto a RWMH con stesso tasso di accettazione.

284.3 · NUTS — No-U-Turn Sampler

Il parametro più critico di HMC è il numero di passi leapfrog L: troppo piccolo e la traiettoria esplora poco, troppo grande e ritorna verso il punto di partenza sprecando valutazioni

del gradiente (il fenomeno U-turn). Il **No-U-Turn Sampler (NUTS)** (Hoffman e Gelman, 2011) elimina la necessità di scegliere L mediante una condizione di arresto automatica: si raddoppia iterativamente la lunghezza della traiettoria (doubling scheme) e si ferma quando il prodotto scalare tra il momento corrente e il displacement dal punto di partenza diventa negativo — segnale che la traiettoria sta tornando indietro. L'algoritmo costruisce un albero binario bilanciato di passi leapfrog in modo time-reversible tramite lo *slice sampling* sul valore dell'Hamiltoniana, e seleziona il punto finale proporzionalmente alla probabilità stazionaria nell'albero. Il *dual averaging* per la taratura automatica di ε al tasso di accettazione ottimale (65%) completa NUTS in un sampler completamente auto-tunante. NUTS è il sampler di default in Stan, PyMC, NumPyro e Turing.jl, ed è lo standard di riferimento per la statistica bayesiana applicata con modelli complessi, avendo reso pratico il campionamento su posteriors di centinaia o migliaia di parametri che erano inaccessibili ai samplers precedenti.

Langevin Dynamics e SGLD: Dal MCMC all'Ottimizzazione Stocastica

Il processo di Langevin come ponte tra campionamento e ottimizzazione — SGLD, SGHMC e Bayesian deep learning scalabile.

285.1 · La Dinamica di Langevin come ODE Stocastica

La **Overdamped Langevin Dynamics** è un processo stocastico continuo ispirato dalla fisica browniana, governato dall'equazione differenziale stocastica (SDE) di Itô:

$$d\theta_t = \frac{1}{2}\nabla_\theta \log P(\theta_t \mid \mathbf{x})\, dt + d\mathbf{W}_t$$

dove $\mathbf{W}_t$ è il moto browniano standard. Il primo termine è una deriva deterministica nella direzione del gradiente della log-posterior (salita verso la moda), e il secondo è un termine di diffusione che garantisce l'esplorazione delle regioni a bassa probabilità. Sotto condizioni di regolarità, questa SDE ha distribuzione stazionaria proporzionale a $e^{\log P(\theta|\mathbf{x})} = P(\theta \mid \mathbf{x})$ — la posterior target. Discretizzando con il metodo di Eulero-Maruyama si ottiene la **Unadjusted Langevin Algorithm (ULA)**: $\theta_{t+1} = \theta_t + \frac{\varepsilon}{2}\nabla_\theta \log P(\theta_t \mid \mathbf{x}) + \sqrt{\varepsilon}\,\boldsymbol{\eta}_t$ con $\boldsymbol{\eta}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. ULA introduce un bias proporzionale a ε (errore di discretizzazione), eliminabile aggiungendo un passo MH correttivo (MALA — Metropolis-Adjusted Langevin Algorithm), a spese di un tasso di accettazione talvolta basso su posteriors non gaussiane.

285.2 · SGLD e il Doppio Ruolo del Rumore Stocastico

Il contributo seminale di Welling e Teh (ICML 2011) è il **Stochastic Gradient Langevin Dynamics (SGLD)**: sostituire il gradiente completo $\nabla_\theta \log P(\theta \mid \mathbf{x})$ con il gradiente stocastico su mini-batch $\tilde{\nabla} = \nabla_\theta \log P(\theta) + \frac{n}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \nabla_\theta \log p(\mathbf{x}_i \mid \theta)$:

$$\theta_{t+1} = \theta_t + \frac{\varepsilon_t}{2} \tilde{\nabla}_\theta \log P(\theta_t \mid \mathbf{x}) + \sqrt{\varepsilon_t}\, \boldsymbol{\eta}_t$$

Il rumore stocastico del gradiente ha varianza $O(\varepsilon_t^2 / |\mathcal{B}|^2)$; il rumore di Langevin ha varianza $O(\varepsilon_t)$. Per $|\mathcal{B}|$ grande e ε_t piccolo, il rumore del gradiente è trascurabile rispetto al rumore di Langevin: SGLD si comporta come ULA con gradiente esatto. Per ε_t che decresce con il programma di Robbins-Monro ($\sum_t \varepsilon_t = \infty$, $\sum_t \varepsilon_t^2 < \infty$), SGLD converge alla vera posterior senza il passo di accettazione MH. Nelle fasi iniziali con passo grande, SGLD si comporta come SGD e converge rapidamente verso la moda; nelle fasi tardive con passo piccolo, esplora la posterior locale intorno alla moda. Questa doppia natura rende SGLD ideale per il Bayesian deep learning: non richiede modifiche all'architettura e si integra naturalmente nelle pipeline di training esistenti, mentre l'ensemble implicito dei parametri campionati produce stime di incertezza più calibrate del frequentismo. Varianti come **SGHMC** (Chen, Fox e Guestrin 2014) aggiungono momenti hamiltoniani per ridurre l'autocorrelazione, e **PSGLD** precondiziona con la Fisher diagonale per adattarsi alla curvatura della posterior.

La Metrica di Fisher e lo Spazio Riemanniano delle Distribuzioni

La struttura geometrica intrinseca sullo spazio dei modelli probabilistici — invarianza alle reparametrizzazioni e unicità di Čencov.

286.1 · Perché la Geometria Euclidea è Inadeguata

L'ottimizzazione di un modello probabilistico $p_\theta(\mathbf{x})$ con gradient descent aggiorna i parametri con passi $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}(\theta)$ che sono sensibili alla parametrizzazione: se si reparametrizza il modello con $\phi = h(\theta)$, la traiettoria di discesa cambia radicalmente anche se la funzione modellata è identica. Questo comportamento è indesiderabile perché la scelta della parametrizzazione è arbitraria — non dovrebbe influenzare l'algoritmo di apprendimento. La **Geometria dell'Informazione**, sistematizzata da Shun-ichi Amari nei suoi lavori dal 1985 (culminati nel libro *Information Geometry and Its Applications*, 2016), dota lo spazio dei parametri di una

struttura riemanniana *intrinseca* che è invariante rispetto alle reparametrizzazioni: la distanza geometrica tra due distribuzioni p_{θ_1} e p_{θ_2} è indipendente dalla parametrizzazione scelta per rappresentarle.

La metrica riemanniana naturale sullo spazio delle distribuzioni è la **Fisher Information Matrix (FIM)**:

$$F_{ij}(\theta) = \mathbb{E}_{p_\theta(\mathbf{x})}\left[\frac{\partial \log p_\theta(\mathbf{x})}{\partial \theta_i}\frac{\partial \log p_\theta(\mathbf{x})}{\partial \theta_j}\right] = -\mathbb{E}_{p_\theta(\mathbf{x})}\left[\frac{\partial^2 \log p_\theta(\mathbf{x})}{\partial \theta_i \partial \theta_j}\right]$$

L'uguaglianza tra le due forme (prodotto delle score functions, e Hessiano atteso del log-likelihood) vale per modelli regolari ed è nota come **identità di Fisher**. Il teorema di unicità di Čencov (1982) stabilisce che la Fisher information è (a meno di costanti scalari positive) l'unica metrica riemanniana sullo spazio delle distribuzioni che sia monotona rispetto alle mappe markoviane — le trasformazioni stocastiche che aggiungono rumore o marginalizzano. Questa unicità conferisce alla FIM uno status privilegiato: non è solo una scelta comoda, è la scelta geometricamente corretta.

286.2 · La KL-Divergenza come Distanza Riemanniana Infinitesima

La connessione tra KL-divergenza e metrica di Fisher è fondamentale per comprendere perché la FIM è la metrica giusta. Per perturbazioni infinitesime $d\theta$:

$$\mathrm{KL}(p_\theta \| p_{\theta+d\theta}) = \frac{1}{2}\, d\theta^T F(\theta)\, d\theta + O(\|d\theta\|^3)$$

La Fisher information è quindi il *Hessiano locale* della KL-divergenza, la misura più naturale di distanza tra distribuzioni. Dal punto di vista statistico, la FIM è anche il limite della precisione raggiungibile per la stima di θ: il **Cramér-Rao bound** $\mathrm{Var}[\hat{\theta}] \geq F(\theta)^{-1}$ afferma che la matrice di covarianza di qualunque stimatore non distorto di θ è almeno uguale all'inversa della Fisher information, e lo stimatore di massima verosimiglianza raggiunge questo bound asintoticamente. La FIM fornisce quindi simultaneamente la metrica geometrica "corretta" per l'ottimizzazione, la misura della curvatura della log-likelihood, e il limite fondamentale della precisione statistica — tre prospettive che convergono sulla stessa quantità.

Natural Gradient di Amari: Il Gradient Descent Geometricamente Corretto

Steepest descent riemanniano, invarianza alle reparametrizzazioni, K-FAC e l'applicazione alle reti neurali.

287.1 · Steepest Descent sotto la Metrica di Fisher

Il gradiente standard $\nabla_\theta \mathcal{L}$ indica la direzione di massima discesa rispetto alla norma euclidea nello spazio dei parametri. Il **Natural Gradient** di Amari (1998) indica invece la direzione di massima discesa rispetto alla norma riemanniana definita dalla Fisher information: dati i vincoli $\|d\theta\|_F^2 = d\theta^T F(\theta)\, d\theta = \varepsilon^2$, la soluzione al problema di ottimizzazione locale $\min_{d\theta:\|d\theta\|_F=\varepsilon} \mathcal{L}(\theta + d\theta)$ è:

$$\tilde{\nabla}_\theta \mathcal{L} = F(\theta)^{-1} \nabla_\theta \mathcal{L}$$

Il natural gradient è il gradiente ordinario precondizionato con l'inversa della Fisher matrix. La sua proprietà definitoria è l'**invarianza alle reparametrizzazioni**: se si parametrizza il modello con $\phi = h(\theta)$, il natural gradient in ϕ è la trasformazione jacobiana del natural gradient in θ — il risultato finale è lo stesso indipendentemente dalla parametrizzazione, a differenza del gradiente ordinario. In termini pratici, questo significa che il natural gradient identifica la "vera" direzione di miglioramento nel modello probabilistico, non dipendente da artefatti della parametrizzazione come la scala o la correlazione tra parametri. Su modelli della famiglia esponenziale, il natural gradient di $-\log P(\mathbf{x} \mid \theta)$ rispetto ai parametri naturali η coincide con l'aggiornamento ottimale del momento — connettendo il natural gradient con l'algoritmo EM in modo elegante.

287.2 · K-FAC: Natural Gradient Scalabile per Reti Neurali

Il costo di calcolo del natural gradient — invertire la Fisher matrix di dimensione $d \times d$ dove d è il numero di parametri — è $O(d^3)$, proibitivo per reti con milioni di parametri. L'algoritmo **Kronecker-Factored Approximate Curvature (K-FAC)** (Martens e Grosse, 2015) rende il natural gradient scalabile approssimando la Fisher matrix come prodotto di Kronecker di matrici molto più piccole, sfruttando la struttura a layer della rete. Per un layer lineare $\mathbf{a} = W\mathbf{s}$ con input $\mathbf{s}$ e segnali di errore $\mathbf{g} = \partial\mathcal{L}/\partial\mathbf{a}$, la Fisher è approssimata come $F_W \approx \hat{A} \otimes \hat{G}$ con $\hat{A} = \mathbb{E}[\mathbf{s}\mathbf{s}^T]$ e $\hat{G} = \mathbb{E}[\mathbf{g}\mathbf{g}^T]$. L'inversa del prodotto di Kronecker è il prodotto degli inversi: $(A \otimes G)^{-1} = A^{-1} \otimes G^{-1}$, riducendo il costo di inversione da $O((d_{\text{in}} d_{\text{out}})^3)$ a $O(d_{\text{in}}^3 + d_{\text{out}}^3)$. K-FAC produce tipicamente convergenze 2-5 volte più rapide in numero di iterazioni rispetto ad

Adam, a spese di un costo per iterazione maggiore, ed è il metodo di riferimento per il training continual learning e per modelli probabilistici complessi come i Bayesian Neural Networks.

Famiglie Esponenziali: Struttura Duale e Statistiche Sufficienti

La classe di distribuzioni con geometria duale naturale — parametri naturali, momenti, log-partition function e cumulanti.

288.1 · Definizione e Proprietà Fondamentali

Le **famiglie esponenziali** sono la classe di distribuzioni che ammette la rappresentazione:

$$p(\mathbf{x} \mid \boldsymbol{\eta}) = h(\mathbf{x}) \exp\big(\boldsymbol{\eta}^T T(\mathbf{x}) - A(\boldsymbol{\eta})\big)$$

dove $\boldsymbol{\eta}$ sono i **parametri naturali**, $T(\mathbf{x})$ le **statistiche sufficienti** (per il teorema di Fisher-Neyman: tutto l'informazione sul parametro $\boldsymbol{\eta}$ contenuta in $\mathbf{x}$ è catturata da $T(\mathbf{x})$), $A(\boldsymbol{\eta}) = \log \int h(\mathbf{x}) e^{\boldsymbol{\eta}^T T(\mathbf{x})} d\mathbf{x}$ è la **log-partition function** (o funzione generatrice dei cumulanti), e $h(\mathbf{x})$ è la misura base. Quasi tutte le distribuzioni comuni appartengono a questa classe: la gaussiana ($T(x) = (x, x^2)$), la Bernoulli ($T(x) = x$), la Poisson ($T(x) = x$), la Gamma ($T(x) = (\log x, x)$), la Dirichlet, la Wishart. La log-partition function genera tutti i cumulanti attraverso la derivazione: $\nabla_{\boldsymbol{\eta}} A = \mathbb{E}[T(\mathbf{x})]$ (momenti del primo ordine), $\nabla^2_{\boldsymbol{\eta}} A = \mathrm{Cov}[T(\mathbf{x})]$ (matrice di covarianza). In particolare, la Fisher Information Matrix è uguale all'Hessiano della log-partition: $F(\boldsymbol{\eta}) = \nabla^2_{\boldsymbol{\eta}} A(\boldsymbol{\eta})$ — un collegamento elegante tra geometria dell'informazione e struttura della famiglia esponenziale.

288.2 · Geometria Duale: Parametri Naturali e Parametri Momento

Ogni famiglia esponenziale ammette due sistemi di coordinate duali: i **parametri naturali** $\boldsymbol{\eta}$ e i **parametri momento** $\boldsymbol{\mu} = \mathbb{E}_{\boldsymbol{\eta}}[T(\mathbf{x})] = \nabla A(\boldsymbol{\eta})$. La trasformazione tra le due coordinate è data dalla dualità di Legendre: $A^*(\boldsymbol{\mu}) = \sup_{\boldsymbol{\eta}}[\boldsymbol{\eta}^T \boldsymbol{\mu} - A(\boldsymbol{\eta})]$, dove $A^*(\boldsymbol{\mu}) = -H[p_{\boldsymbol{\mu}}]$ è la negativa dell'entropia della distribuzione con momenti $\boldsymbol{\mu}$. La KL-divergenza tra due membri della stessa famiglia esponenziale si esprime come **divergenza di Bregman** associata ad A: $\mathrm{KL}(p_{\boldsymbol{\mu}} \| p_{\boldsymbol{\eta}}) = A(\boldsymbol{\eta}) - A(\boldsymbol{\eta}') - (\boldsymbol{\mu} - \boldsymbol{\mu}')^T \boldsymbol{\eta}'$. Questa dualità ha conseguenze profonde per l'algoritmo EM: il passo E, che calcola l'aspettativa delle statistiche sufficienti sotto la posterior corrente, è una proiezione nello spazio dei momenti sulla *m-flat manifold* del parametro corrente; il passo M,

che massimizza la likelihood con i momenti fissi, è una proiezione nello spazio dei parametri naturali sulla *e-flat manifold*. La convergenza dell'EM è quindi geometricamente una proiezione alternata di Csiszár, che garantisce la convergenza al punto di intersezione delle due varietà duali.

α-Divergenze: Una Famiglia Parametrica tra KL e Hellinger

La generalizzazione continua della KL-divergenza — tradeoff mode-seeking vs mass-covering nei modelli generativi.

289.1 · Definizione e Casi Speciali

Le **α-divergenze**, introdotte da Amari nel contesto della geometria dell'informazione, generalizzano la KL-divergenza in una famiglia continua parametrizzata da $\alpha \in \mathbb{R}$:

$$D_\alpha(p\|q) = \frac{4}{1-\alpha^2}\left(1 - \int p(\mathbf{x})^{\frac{1+\alpha}{2}} q(\mathbf{x})^{\frac{1-\alpha}{2}} d\mathbf{x}\right)$$

I limiti $\alpha \to +1$ e $\alpha \to -1$ recuperano $\mathrm{KL}(p\|q)$ e $\mathrm{KL}(q\|p)$ rispettivamente. Per $\alpha = 0$ si ottiene la **divergenza di Hellinger** $H^2(p,q) = \int (\sqrt{p} - \sqrt{q})^2 \, d\mathbf{x}$ — simmetrica e metrica (soddisfa la disuguaglianza triangolare, a differenza della KL). Per $\alpha = -1$ si ha la χ^2-divergenza. Tutte le α-divergenze appartengono alla classe delle *f-divergenze* e sono invarianti alle trasformazioni invertibili dello spazio di campionamento — proprietà che le rende candidate naturali per misurare la dissimilarità tra distribuzioni in modo indipendente dalla rappresentazione.

289.2 · Il Tradeoff Mode-Seeking vs Mass-Covering e le GAN

Il comportamento delle α-divergenze rispetto alle distribuzioni multimodali rivela un tradeoff fondamentale che ha implicazioni dirette per il design dei modelli generativi. Per α positivo, la divergenza penalizza $q(x) \approx 0$ dove $p(x) > 0$, producendo soluzioni *mass-covering* (zero-avoiding) che cercano di coprire tutti i modi della distribuzione target. Per α negativo (in particolare $\mathrm{KL}(q\|p)$ per $\alpha \to -1$), la divergenza è insensibile a $q(x) = 0$ dove $p(x) > 0$ ma penalizza $q(x) > 0$ dove $p(x) = 0$, producendo soluzioni *mode-seeking* (zero-forcing) concentrate su un singolo modo. La divergenza di Jensen-Shannon $D_{JS}(p\|q) = \frac{1}{2}\mathrm{KL}(p\|m) + \frac{1}{2}\mathrm{KL}(q\|m)$ con $m = (p+q)/2$, corrispondente ad $\alpha = 0$, è la loss delle GAN vanilla (Goodfellow et al. 2014) nel limite ottimale del discriminatore. La **Wasserstein GAN** sostituisce la JS-divergenza con

la distanza di Wasserstein $W_1(p,q) = \inf_{\gamma \in \Pi(p,q)} \mathbb{E}_{(x,y)\sim\gamma}[\|x-y\|]$, che è definita anche quando p e q hanno supporti disgiunti (dove la JS-divergenza satura a $\log 2$) e fornisce gradienti più informativi durante il training. Il Flow Matching (§298) usa invece la distanza L^2 tra traiettorie come loss — un'ulteriore alternativa con migliori proprietà di training.

AdaGrad, Adam e la Connessione con il Natural Gradient

Gli ottimizzatori adattativi moderni come approssimazioni del natural gradient — Fisher diagonale, RMSProp, Adam.

290.1 · La Fisher Diagonale e AdaGrad

Il natural gradient pieno $F(\theta)^{-1}\nabla_\theta \mathcal{L}$ è computazionalmente inaccessibile per reti profonde. Gli ottimizzatori adattativi moderni possono essere interpretati come approssimazioni del natural gradient che usano stime diagonali della Fisher matrix, aggiornate online tramite media dei gradienti al quadrato. **AdaGrad** (Duchi, Hazan e Singer, JMLR 2011) accumula il quadrato dei gradienti storici per scalare adattativamente il learning rate per ogni coordinata:

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \varepsilon}} \nabla_{\theta_i} \mathcal{L}_t, \qquad G_{t,ii} = \sum_{\tau=1}^{t} (\nabla_{\theta_i} \mathcal{L}_\tau)^2$$

La connessione con la geometria dell'informazione è la seguente: la diagonale della Fisher information è $F_{ii}(\theta) = \mathbb{E}[(\partial \log p_\theta / \partial \theta_i)^2]$, esattamente la varianza della score function nella direzione i. AdaGrad stima F_{ii} tramite la media temporale del gradiente al quadrato — uno stimatore consistente quando la loss è la log-likelihood. Il natural gradient diagonale $\tilde{\nabla}_{\theta_i}\mathcal{L} = \nabla_{\theta_i}\mathcal{L} / F_{ii}(\theta)$ ha la stessa forma dell'aggiornamento di AdaGrad, con $G_{t,ii}$ che stima F_{ii}. Questo spiega perché AdaGrad funziona particolarmente bene per parametri sparsi come i word embeddings: per parole rare, la Fisher diagonale è bassa (il parametro porta poco segnale in media), e AdaGrad compensa con un learning rate effettivo più alto — esattamente come farebbe il natural gradient.

290.2 · Adam come Natural Gradient con Momento e Finestra Temporale

Adam (Kingma e Ba, ICLR 2015) estende AdaGrad con due innovazioni: un primo momento (velocità) $m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla\mathcal{L}_t$ e una media mobile esponenziale del secondo momento $v_t = \beta_2 v_{t-1} + (1-\beta_2)(\nabla\mathcal{L}_t)^2$, corretti per il bias iniziale: $\hat{m}_t = m_t/(1-\beta_1^t)$, $\hat{v}_t = v_t/(1-$

β_2^t). L'aggiornamento $\theta_{t+1} = \theta_t - \eta\hat{m}_t/\sqrt{\hat{v}_t + \varepsilon}$ ha un'interpretazione naturale: $\sqrt{\hat{v}_t}$ è una stima della radice quadrata della Fisher diagonale con finestra temporale di circa $1/(1-\beta_2)$ iterazioni (tipicamente 1000 con $\beta_2 = 0.999$), e $\hat{m}_t$ è il gradiente naturale smussato con inerzia. La bias correction è fondamentale nelle prime iterazioni quando m_t e v_t sono sottostimati rispetto al loro valore stazionario: senza correzione, i primi aggiornamenti sarebbero quasi nulli indipendentemente dal segnale del gradiente. Il valore ottimale del learning rate per Adam è notoriamente più robusto rispetto a SGD — conseguenza della precondizionatura adattativa che normalizza automaticamente l'eterogeneità delle scale dei gradienti in diverse parti della rete.

Geometria Duale di Amari e l'Algoritmo EM come Proiezione Alternata

Connessioni e-flat e m-flat — il teorema di proiezione di Amari e la convergenza geometrica dell'EM.

291.1 · Le Due Connessioni Affini e la Doppia Geometria

Lo spazio delle distribuzioni nella geometria dell'informazione non ha una singola connessione affine ma una coppia di **connessioni duali** $\nabla^{(e)}$ e $\nabla^{(m)}$, duali rispetto alla metrica di Fisher nel senso che $\partial_k g_{ij} = \Gamma^{(e)}_{k,ij} + \Gamma^{(m)}_{k,ji}$. Una varietà è **e-flat** (o exponential-flat) se le geodetiche secondo $\nabla^{(e)}$ sono rette nello spazio dei parametri naturali $\boldsymbol{\eta}$ — le famiglie esponenziali sono e-flat per costruzione. Una varietà è **m-flat** (o mixture-flat) se le geodetiche secondo $\nabla^{(m)}$ sono rette nello spazio dei parametri momento $\boldsymbol{\mu}$ — le miscele convesse di distribuzioni fisse sono m-flat. La Fisher information è la metrica riemanniana che rende queste due connessioni doppie nel senso di Amari: $(\mathcal{S}, g, \nabla^{(e)}, \nabla^{(m)})$ è una varietà doppie-piatta, la struttura geometrica più ricca dopo quella piatta.

Il **Teorema di Proiezione di Amari** afferma che la proiezione di un punto p su una sotto-varietà e-flat $\mathcal{M}_e$ (misurata in $\mathrm{KL}(p\|q)$, la KL "in avanti") è il piede della perpendicolare m-geodetica passante per p rispetto a $\mathcal{M}_e$, e il piede è il punto dove le due geodetiche si intersecano ortogonalmente rispetto alla metrica di Fisher. Il risultato duale vale per le proiezioni m-flat misurate in $\mathrm{KL}(q\|p)$. L'algoritmo EM è l'instanza più famosa di questa struttura: il passo E è la proiezione m-ortogonale sul sottospazio delle distribuzioni con momenti fissi (m-flat manifold parametrizzata da $\boldsymbol{\theta}$ fisso), e il passo M è la proiezione e-ortogonale sul sottospazio e-flat delle distribuzioni del modello parametrizzato. Ogni iterazione dell'EM diminuisce monotonamente la KL tra la distribuzione corrente e quella ottimale, convergendo al punto di intersezione delle

due varietà duali — il massimo della likelihood sotto il modello parametrico. Questo framework geometrico non solo spiega la convergenza dell'EM, ma suggerisce anche le sue naturali generalizzazioni a setting non coniugati tramite proiezioni approssimate.

Neural ODE e l'Adjoint Method Completo

Reti neurali come sistemi dinamici continui — calcolo efficiente del gradiente con memoria costante tramite l'ODE aggiunta.

292.1 · Dalle ResNet alle ODE Continue

L'analogia tra ResNet e integratori numerici è stata osservata da diversi autori (He et al. 2015, Haber e Ruthotto 2017, Lu et al. 2017), ma è con il lavoro di Chen, Rubanova, Bettencourt e Duvenaud (NeurIPS 2018) che l'idea è stata formalizzata in un modello autonomo e differenziabile: i **Neural Ordinary Differential Equations**. Una ResNet con layer uniformi $\mathbf{h}_{t+1} = \mathbf{h}_t + f(\mathbf{h}_t, \theta_t)$ è l'integrazione di Eulero di passo unitario dell'ODE $d\mathbf{h}/dt = f(\mathbf{h}(t), \theta)$. I Neural ODE abbracciano questa interpretazione: la rete è definita direttamente come una ODE con dynamics parametrizzate da f_θ, e l'output è ottenuto integrando numericamente l'ODE da t_0 a T:

$$\mathbf{h}(T) = \mathbf{h}(t_0) + \int_{t_0}^{T} f(\mathbf{h}(t), \theta)\, dt = \text{ODESolve}(\mathbf{h}_0, f_\theta, t_0, T)$$

Il solutore adattativo (tipicamente Dormand-Prince RK45 o Tsit5) sceglie automaticamente il passo di integrazione per soddisfare una tolleranza di errore relativa e assoluta specificata, adattando la "profondità effettiva" della rete alla difficoltà dell'input. I vantaggi rispetto alle ResNet sono tre: *profondità adattiva* senza specificare il numero di layer; *memoria costante* durante il forward pass poiché non si memorizzano le attivazioni intermedie (la traiettoria viene rivalutata se necessaria durante il backward); e una formulazione naturale per *dati irregolarmente campionati nel tempo* (time series con osservazioni mancanti, ODE stochastiche), poiché il solutore può essere valutato in qualunque punto temporale senza interpolazione.

292.2 · L'Adjoint Method: Derivazione Completa

Il calcolo di $d\mathcal{L}/d\theta$ per un Neural ODE non può utilizzare la backpropagation standard attraverso le operazioni interne del solutore ODE — questo richiederebbe di memorizzare tutte

le attivazioni intermedie lungo la traiettoria, vanificando il vantaggio di memoria. L'**adjoint method**, derivato dalla teoria del controllo ottimale di Pontryagin (1962), calcola il gradiente integrando all'indietro nel tempo un'ODE ausiliaria — l'*ODE aggiunta* — con costo di memoria $O(1)$ rispetto alla profondità della traiettoria.

DERIVAZIONE COMPLETA — Adjoint Method

Definire il Lagrangiano aumentato $\tilde{\mathcal{L}} = \mathcal{L}(\mathbf{h}(T)) + \int_{t_0}^{T} \mathbf{a}(t)^T \big(\frac{d\mathbf{h}}{dt} - f(\mathbf{h}(t), \theta)\big)\, dt$, identicamente uguale a $\mathcal{L}$ per traiettorie che soddisfano l'ODE. I moltiplicatori $\mathbf{a}(t)$ sono le variabili di costate (stato aggiunto). Principio dei moltiplicatori di Lagrange per vincoli ODE

Calcolare la variazione di $\tilde{\mathcal{L}}$ rispetto alla perturbazione $\delta\mathbf{h}(t)$ della traiettoria. Il termine con $d(\delta\mathbf{h})/dt$ si integra per parti: $\int \mathbf{a}^T \frac{d\delta\mathbf{h}}{dt}\, dt = [\mathbf{a}^T \delta\mathbf{h}]_{t_0}^{T} - \int \frac{d\mathbf{a}^T}{dt} \delta\mathbf{h}\, dt$. Integrazione per parti per eliminare la derivata di $\delta\mathbf{h}$

Imponendo che il coefficiente di $\delta\mathbf{h}(t)$ sia zero per ogni $t \in (t_0, T)$, si ottiene l'**ODE aggiunta** con condizione terminale $\mathbf{a}(T) = \partial\mathcal{L}/\partial\mathbf{h}(T)$:

$$\frac{d\mathbf{a}}{dt} = -\mathbf{a}(t)^T \frac{\partial f(\mathbf{h}(t), \theta)}{\partial \mathbf{h}}$$

Equazione di Euler-Lagrange per lo stato aggiunto

Il gradiente rispetto ai parametri si ottiene prendendo la variazione rispetto a θ:

$$\frac{d\mathcal{L}}{d\theta} = -\int_{T}^{t_0} \mathbf{a}(t)^T \frac{\partial f(\mathbf{h}(t), \theta)}{\partial \theta}\, dt$$

L'integrale del prodotto stato aggiunto–Jacobiano rispetto a θ

In pratica, le tre quantità $(\mathbf{h}(t), \mathbf{a}(t), d\mathcal{L}/d\theta)$ vengono calcolate simultaneamente integrando all'indietro da T a t_0 l'ODE aumentata di dimensione $\dim(\mathbf{h}) + \dim(\mathbf{h}) + \dim(\theta)$, chiamando f_θ e il suo Jacobiano solo al tempo corrente. Implementazione con singola ODE backward di dimensione aumentata

L'implementazione pratica (torchdiffeq, diffrax) usa il checkpointing per la rivalutazione di $\mathbf{h}(t)$ durante il backward pass: invece di memorizzare la traiettoria completa, si memorizzano K checkpoint e si rivaluta f_θ tra checkpoint con il solutore forward, limitando il costo di memoria a $O(K \dim \mathbf{h})$ anziché $O(L \dim \mathbf{h})$ con L passi del solutore.

Continuous Normalizing Flows e FFJORD

Cambiamento di variabile continuo — l'equazione di continuità, lo stimatore di Hutchinson e i modelli generativi con ODE.

293.1 · L'Equazione di Continuità per la Log-Densità

I **Continuous Normalizing Flows (CNF)** usano una Neural ODE come trasformazione generativa: si parte da $\mathbf{z}(0) \sim p_0$ (tipicamente gaussiana) e si integra $d\mathbf{z}/dt = f(\mathbf{z}(t), t; \theta)$ da $t = 0$ a $t = 1$ per ottenere $\mathbf{z}(1) \sim p_1$ (distribuzione target). Il teorema di cambiamento di variabile per flussi continui — derivato dall'equazione di continuità $\partial_t p + \nabla \cdot (p\, f) = 0$ — afferma che la log-densità evolve come:

$$\frac{d \log p(\mathbf{z}(t))}{dt} = -\mathrm{tr}\left(\frac{\partial f(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)}\right)$$

Il problema è che il calcolo esatto di $\mathrm{tr}(\partial f/\partial \mathbf{z})$ richiede $O(d)$ valutazioni di f (una per riga del Jacobiano). **FFJORD** (Grathwohl, Chen, Bettencourt, Sutskever e Duvenaud, ICLR 2019) usa lo **stimatore di Hutchinson** per stimare la traccia in modo non-biasato con una singola forward-backward pass: per qualunque vettore $\boldsymbol{\epsilon}$ con $\mathbb{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T] = I$, si ha $\mathrm{tr}(A) = \mathbb{E}[\boldsymbol{\epsilon}^T A \boldsymbol{\epsilon}]$. Il prodotto $\boldsymbol{\epsilon}^T(\partial f/\partial \mathbf{z})$ è calcolabile con una singola backward pass rispetto a $\mathbf{z}$ del valore scalare $\boldsymbol{\epsilon}^T f(\mathbf{z})$ — riducendo il costo a $O(1)$ valutazioni di f per la stima della traccia, rendendo i CNF scalabili a dimensioni dell'ordine di migliaia.

293.2 · Confronto con i Normalizing Flows Discreti e Applicazioni

Prima dei Continuous Normalizing Flows, i modelli generativi a flusso erano costruiti come composizioni di trasformazioni invertibili discrete — RealNVP (Dinh, Sohl-Dickstein e Bengio 2016), Glow (Kingma e Dhariwal 2018) — dove ciascuna trasformazione ha una struttura speciale che rende il determinante dello Jacobiano calcolabile in tempo lineare. RealNVP, ad esempio, divide le dimensioni in due blocchi: le prime $d/2$ passano invariate, le seconde vengono scalate e traslate in funzione delle prime. Il determinante risultante è triangolare a blocchi e quindi calcolabile in $O(d)$ anziché $O(d^3)$. Il vincolo architetturale è però stringente: non si possono usare strati arbitrari che mescolino liberamente tutte le dimensioni, e la rappresentatività del modello dipende dall'ordine con cui si alternano i blocchi in layer successivi. Ogni aggiunta di espressività (convoluzioni $1 times 1$ in Glow, trasformazioni polinomiali in Neural Spline Flows) richiede una riprogettazione ad hoc della struttura invertibile.

I CNF con FFJORD superano questo vincolo strutturale nella sua interezza: il campo vettoriale $f(\mathbf{z}, t; heta)$ può essere una qualunque rete neurale arbitraria — trasformers, convoluzioni, GNN — senza nessun vincolo di invertibilità esplicita, poiché l'invertibilità è garantita dall'unicità della soluzione dell'ODE sotto le condizioni di Picard-Lindelöf. Questa libertà architettonica ha permesso l'applicazione dei CNF a domini prima inaccessibili ai flow discreti: la generazione di molecole tridimensionali con equivarianza $SE(3)$ (Equivariant Flows), la stima di densità su varietà non euclidee come tori e sfere (Riemannian Flows), e la modellazione di dati di forma biologica 3D.

Hamiltonian Neural Networks: Conservazione dell'Energia per Architettura

Apprendimento di sistemi conservativi — la struttura hamiltoniana come vincolo architetturale, generalizzazione fuori distribuzione.

294.1 · Il Problema della Deriva Energetica nelle Reti Standard

Addestrare una rete neurale a predire le traiettorie di un sistema fisico conservativo (pendolo, orbite planetarie, dinamica molecolare) senza imporre la struttura fisica porta inevitabilmente a una *deriva energetica*: la rete minimizza l'errore quadratico medio sulla traiettoria di training, ma le sue predizioni violano la conservazione dell'energia per tempi lunghi — un'infedeltà fisicamente inaccettabile che peggiora esponenzialmente con il tempo di simulazione. Le **Hamiltonian Neural Networks (HNN)** (Greydanus, Dzamba e Yosinski, NeurIPS 2019) risolvono questo problema fondamentalmente cambiando cosa viene appreso: invece di predire direttamente $d\mathbf{q}/dt$ e $d\mathbf{p}/dt$, si apprende una funzione scalare $\mathcal{H}_\theta(\mathbf{q}, \mathbf{p})$ che rappresenta l'Hamiltoniana del sistema, e le equazioni del moto sono derivate dalla struttura hamiltoniana:

$$\frac{d\mathbf{q}}{dt} = \frac{\partial \mathcal{H}_\theta}{\partial \mathbf{p}}, \qquad \frac{d\mathbf{p}}{dt} = -\frac{\partial \mathcal{H}_\theta}{\partial \mathbf{q}}$$

Per le equazioni di Hamilton, $d\mathcal{H}/dt = (\partial\mathcal{H}/\partial\mathbf{q})\dot{\mathbf{q}} + (\partial\mathcal{H}/\partial\mathbf{p})\dot{\mathbf{p}} = (\partial\mathcal{H}/\partial\mathbf{q})(\partial\mathcal{H}/\partial\mathbf{p}) - (\partial\mathcal{H}/\partial\mathbf{p})(\partial\mathcal{H}/\partial\mathbf{q}) = 0$ identicamente, per qualunque valore dei pesi θ. La conservazione dell'energia è quindi garantita strutturalmente dall'architettura, non dal training. Le HNN generalizzano sistematicamente meglio fuori dalla distribuzione di training rispetto alle reti standard: per un pendolo addestrato su traiettorie di breve durata, le HNN mantengono l'accuratezza per tempi di simulazione 10-100 volte più lunghi, poiché la conservazione dell'energia limita lo spazio delle

traiettorie accessibili a quelle fisicamente plausibili.

Lagrangian Neural Networks: Meccanica Analitica e Coordinate Generalizzate

Apprendimento del Lagrangiano — equazioni di Eulero-Lagrange, vincoli geometrici e trasformata di Legendre neurale.

295.1 · Il Formalismo Lagrangiano per Sistemi Vincolati

Mentre le HNN operano nello spazio delle fasi $(\mathbf{q}, \mathbf{p})$, molti sistemi fisici hanno vincoli geometrici che rendono naturale il formalismo lagrangiano: il pendolo sferico è descritto da due angoli (non da tre coordinate cartesiane con il vincolo della lunghezza del filo), la dinamica del braccio robotico è descritta dagli angoli dei giunti, la conformazione delle proteine è descritta dagli angoli diedrali. Le **Lagrangian Neural Networks (LNN)** (Cranmer, Greydanus e Ho, ICLR 2020) apprendono la funzione di Lagrange $\mathcal{L}_\theta(\mathbf{q}, \dot{\mathbf{q}})$ e derivano le equazioni del moto tramite le equazioni di Eulero-Lagrange:

$$\frac{d}{dt}\frac{\partial \mathcal{L}_\theta}{\partial \dot{\mathbf{q}}} - \frac{\partial \mathcal{L}_\theta}{\partial \mathbf{q}} = \mathbf{0}, \quad \Rightarrow \quad \ddot{\mathbf{q}} = \left(\nabla^2_{\dot{\mathbf{q}}\dot{\mathbf{q}}}\mathcal{L}_\theta\right)^{-1}\left(\nabla_{\mathbf{q}}\mathcal{L}_\theta - \nabla^2_{\dot{\mathbf{q}}\mathbf{q}}\mathcal{L}_\theta\,\dot{\mathbf{q}}\right)$$

L'inversione della matrice di massa generalizzata $M(\mathbf{q}) = \nabla^2_{\dot{\mathbf{q}}\dot{\mathbf{q}}}\mathcal{L}_\theta$ (il Hessiano del Lagrangiano rispetto alle velocità generalizzate) è il collo di bottiglia computazionale: deve essere positiva definita per garantire l'esistenza della soluzione, e alcune architetture la impongono per costruzione decomponendola come $M(\mathbf{q}) = L(\mathbf{q})L(\mathbf{q})^T$ con L triangolare inferiore. La connessione Lagrangiano-Hamiltoniano passa per la trasformata di Legendre $\mathcal{H}(\mathbf{q}, \mathbf{p}) = \mathbf{p}^T\dot{\mathbf{q}} - \mathcal{L}_\theta(\mathbf{q}, \dot{\mathbf{q}})$ con $\mathbf{p} = \partial\mathcal{L}_\theta/\partial\dot{\mathbf{q}}$: le LNN si convertono automaticamente in HNN tramite questa trasformazione, unificando i due formalismi in una stessa architettura.

Physics-Informed Neural Networks: PDE come Vincoli di Regolarizzazione

La rete neurale come ansatz per equazioni differenziali — residui, problemi inversi e data assimilation.

296.1 · L'Approccio Mesh-Free alle PDE

La simulazione numerica classica di equazioni differenziali parziali (PDE) richiede la discretizzazione del dominio in una mesh — griglia di elementi finiti, volumi finiti o differenze finite — con milioni di nodi per problemi tridimensionali con geometrie complesse. I **Physics-Informed Neural Networks (PINN)** (Raissi, Perdikaris e Karniadakis, Journal of Computational Physics 2019) propongono un paradigma radicalmente diverso: la soluzione della PDE è rappresentata da una rete neurale $u_\theta(\mathbf{x}, t)$ i cui parametri vengono ottimizzati per soddisfare simultaneamente la PDE, le condizioni al contorno e i dati osservati. Per la PDE generica $\mathcal{N}[u](\mathbf{x}, t) = 0$ nel dominio Ω con condizioni di Dirichlet $u = g$ su $\partial\Omega$, la loss multioggettivo è:

$$\mathcal{L}(\theta) = \frac{w_r}{N_r}\sum_{i=1}^{N_r} |\mathcal{N}[u_\theta](\mathbf{x}_i^r, t_i^r)|^2 + \frac{w_b}{N_b}\sum_{j=1}^{N_b} \left|u_\theta(\mathbf{x}_j^b, t_j^b) - g(\mathbf{x}_j^b, t_j^b)\right|^2$$

I punti di collocazione $\{(\mathbf{x}_i^r, t_i^r)\}$ per il residuo PDE sono campionati nel dominio con qualunque strategia (uniforme, importanza, Latin hypercube); le derivate $\mathcal{N}[u_\theta]$ sono calcolate tramite *automatic differentiation* rispetto agli input della rete. L'approccio è intrinsecamente *mesh-free*: la rete valuta la soluzione in qualunque punto del dominio continuo senza richiedere una discretizzazione, ed è particolarmente efficiente per domini ad alta dimensione dove le mesh classiche crescono esponenzialmente nel numero di nodi. Il prezzo è la dipendenza dalla qualità dell'ottimizzazione (convergenza a minimi locali, sensibilità ai punti di collocazione) e la competitività computazionale rispetto a metodi classici per problemi 1D-3D standard dove le mesh sono efficienti.

296.2 · Problemi Inversi con PINN

La versatilità dei PINN emerge pienamente nei **problemi inversi**: identificare i parametri della PDE (viscosità, diffusività, costanti elastiche) a partire da osservazioni sparse della soluzione. I parametri della PDE — ad esempio la viscosità ν nell'equazione di Navier-Stokes $\partial_t \mathbf{u} + (\mathbf{u}\cdot\nabla)\mathbf{u} = -\nabla p + \nu\Delta\mathbf{u}$ — diventano variabili aggiuntive ottimizzate congiuntamente ai pesi θ della rete, con il residuo della PDE come vincolo di regolarizzazione. La loss diventa $\mathcal{L}(\theta, \nu) = \mathcal{L}_{\text{PDE}}(\theta, \nu) + \mathcal{L}_{\text{dati}}(\theta)$, dove il secondo termine confronta le predizioni della rete con le osservazioni sperimentali. Raissi et al. hanno dimostrato la stima accurata della viscosità nell'equazione di Burgers da dati sparsi con errori dell'ordine del 10^{-4}, confrontabili con i metodi variazionali classici ma senza richiedere la specifica esplicita del modello per ogni istanza del problema.

DeepONet, Neural Operators e il Framework SciML

Apprendimento di operatori tra spazi di funzioni — il teorema universale e le architetture branch-trunk.

297.1 · Dal Punto alla Funzione: Operatori Neurali

I PINN apprendono una soluzione specifica per condizioni al contorno e parametri fissati. Ogni cambiamento nelle condizioni iniziali richiede un nuovo processo di ottimizzazione. I **Neural Operators** vanno oltre: apprendono l'*operatore soluzione* $\mathcal{G}^+ : u_0 \mapsto u(\cdot, T)$, la mappa funzionale dalle condizioni iniziali all'intera soluzione, applicabile a nuove condizioni iniziali con un singolo forward pass. **DeepONet** (Lu, Jin, Pang, Zhang e Karniadakis, Nature Machine Intelligence 2021) è fondato sul Teorema Universale degli Operatori (Chen e Chen 1995): qualunque operatore nonlineare continuo $\mathcal{G}$ può essere approssimato come $\mathcal{G}(u)(y) \approx \sum_{k=1}^{p} b_k(u(x_1), \ldots, u(x_m)) \cdot t_k(y)$, dove la **branch network** $b_k(\cdot)$ codifica la funzione di input u campionata in m punti fissi, e la **trunk network** $t_k(\cdot)$ valuta la base delle funzioni di output nel punto query y. Il prodotto interno dei due output produce la valutazione dell'operatore applicato. DeepONet è addestrato su coppie $\{(u^{(s)}(\cdot), \mathcal{G}(u^{(s)})(\cdot))\}_{s=1}^{S}$ con campionamento casuale dei punti di valutazione dell'output, generalizza a nuove funzioni di input, e produce errori di generalizzazione tipicamente inferiori al 1% per operatori di PDE standard con training su poche migliaia di esempi.

297.2 · Il Framework SciML e la Programmazione Differenziabile per la Scienza

Il termine **Scientific Machine Learning (SciML)** denota l'integrazione sistematica di metodi di ML con la conoscenza delle leggi fisiche, in un ecosistema di programmazione differenziabile che permette di ottimizzare qualunque pipeline che include simulazioni fisiche. L'ecosistema SciML in Julia (Rackauckas e Nie 2017, e successori) fornisce solutori di ODE/PDE/SDE differenziabili attraverso l'adjoint method, Neural ODE, PINN, Neural Operators, e sensitivity analysis continua, il tutto componibile con librerie di ML standard. La differenziabilità end-to-end permette applicazioni prima impraticabili: la progettazione di forme aerodinamiche ottimizzando la geometria rispetto alla portanza calcolata da Navier-Stokes, la sintesi di molecole con proprietà termodinamiche target dove la loss è l'energia libera calcolata da simulazioni MD, e la calibrazione di modelli climatici rispetto a dati satellitari dove il forward model è un sistema di PDE non lineari. La comunità SciML ha dimostrato accelerazioni di ordini di grandezza rispetto ai metodi di ottimizzazione classici black-box su problemi di ingegneria complessa.

Flow Matching: Trasporto Ottimale Continuo per la Generazione

Un'alternativa scalabile ai modelli di diffusione — percorsi lineari, campo vettoriale condizionale e Bridge Matching.

298.1 · Conditional Flow Matching e Percorsi Lineari

I modelli di diffusione (DDPM, Score Matching) addestrano una rete a invertire un processo di corruzione gaussiana progressiva: il training richiede la simulazione del processo di diffusione forward per calcolare il target di denoising, e il campionamento richiede centinaia o migliaia di passi di integrazione. Il **Flow Matching** (Lipman, Chen, Ben-Hamu, Nickel e Le, ICLR 2023) propone un framework concettualmente più semplice: si definisce un percorso deterministico $\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\mathbf{x}_1$ che interpola linearmente dalla distribuzione sorgente $\mathbf{x}_0 \sim p_0$ alla distribuzione target $\mathbf{x}_1 \sim p_1$, e si addestra la rete a regressione sul campo vettoriale che genera questo percorso:

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t,\mathbf{x}_0,\mathbf{x}_1}\left[\|v_\theta(\mathbf{x}_t, t) - (\mathbf{x}_1 - \mathbf{x}_0)\|^2\right]$$

Il campo vettoriale target $u_t(\mathbf{x}_t \mid \mathbf{x}_0, \mathbf{x}_1) = \mathbf{x}_1 - \mathbf{x}_0$ è costante lungo ogni percorso e indipendente da t — una proprietà che produce traiettorie quasi rettilinee, riducendo drasticamente il numero di passi necessari per l'integrazione al campionamento (tipicamente 10-50 contro i 1000 della diffusione). Il Conditional Flow Matching è la versione pratica: si condiziona sul punto di arrivo $\mathbf{x}_1$ per costruire un campo vettoriale condizionale facilmente calcolabile, e si marginalizza sul rumore di sorgente per ottenere il campo vettoriale marginale che guida p_0 verso p_1. Il framework è flessibile: la scelta del percorso di interpolazione (lineare, geodetico ottimale per il trasporto ottimale, percorso di diffusione) determina le proprietà del campo vettoriale appreso e la qualità dei campioni generati. Il **Bridge Matching** estende il framework a percorsi condizionati su coppie specifiche di punti, aprendo applicazioni all'image-to-image translation, alla drug discovery condizionale e all'alignment di sequenze biologiche.

Structural Causal Models di Pearl: Variabili, Funzioni e Meccanismi

La formalizzazione matematica della causalità — come separare correlazione da causazione con strumenti algebrici rigorosi.

299.1 · Il Limite della Statistica Osservazionale

La statistica classica, incluso il machine learning, è fondamentalmente osservazionale: dato un dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}$, si apprende la distribuzione condizionale $P(Y \mid X)$. Questa quantità risponde a domande del tipo "Se osservo $X = x$, qual è la distribuzione di Y?" — ma non a domande del tipo "Se *forzo* $X = x$, quale sarà la distribuzione di Y?" Le due domande hanno risposte diverse in presenza di variabili confondenti: la prima può essere confusa da fattori che influenzano sia X che Y, la seconda isola l'effetto diretto di X su Y. Judea Pearl, nel suo lavoro fondativo culminato nel libro *Causality* (2000, seconda edizione 2009), ha costruito un framework matematico completo per ragionare causalmente: gli **Structural Causal Models (SCM)**.

Uno SCM $\mathcal{M} = (\mathbf{U}, \mathbf{V}, \mathbf{F}, P(\mathbf{U}))$ è definito da: variabili **esogene** $\mathbf{U} = \{U_1, \ldots, U_m\}$ (disturbi, errori, fattori latenti non modellati); variabili **endogene** $\mathbf{V} = \{V_1, \ldots, V_n\}$ (le variabili del sistema osservate); **funzioni strutturali** $\mathbf{F} = \{f_1, \ldots, f_n\}$ dove $V_i = f_i(\text{pa}(V_i), U_i)$ determina ogni variabile endogena dai suoi genitori causali e dal proprio rumore idiosincratico; e la distribuzione $P(\mathbf{U})$ sulle variabili esogene, tipicamente con componenti indipendenti. Le funzioni strutturali non sono semplici regressioni condizionali: esse codificano il *meccanismo causale* invariante che genera V_i e che rimane inalterato quando si interviene su altre variabili del sistema. Lo stesso dataset osservazionale $\{(\mathbf{v}_1, \ldots, \mathbf{v}_n)^{(i)}\}$ può essere generato da SCM con strutture causali radicalmente diverse — il grafo causale non è identificabile dai soli dati osservazionali senza assunzioni aggiuntive.

299.2 · L'Intervento come Chirurgia del Grafo

La potenza degli SCM emerge nella trattazione degli interventi. L'operatore $do(X = x)$ è formalmente implementato come la sostituzione della funzione strutturale di X con la costante x: $X = f_X(\text{pa}(X), U_X) \to X = x$. Geometricamente, questo corrisponde alla *chirurgia del grafo causale*: si rimuovono tutti gli archi entranti in X nel DAG $\mathcal{G}$, ottenendo il grafo mutilato $\mathcal{G}_{\overline{X}}$, e si propaga il valore $X = x$ attraverso le funzioni strutturali degli altri nodi. La distribuzione post-intervento $P(\mathbf{V} \mid do(X = x))$ — diversa dalla distribuzione condizionale osservazionale $P(\mathbf{V} \mid X = x)$ in presenza di confondenti — è la quantità causale fondamentale. L'uguaglianza $P(\mathbf{V} \mid do(X = x)) = P(\mathbf{V} \mid X = x)$ vale se e solo se X è un *assegnazione causale* nel senso che non ha cause comuni con Y nel grafo — le condizioni formali di questa uguaglianza sono fornite dal do-calculus della §300.

Do-Calculus: Le Tre Regole per l'Identificabilità Causale

Un sistema assiomatico completo per trasformare espressioni causali in osservazionali — aggiustamento backdoor, frontdoor e criteri grafici.

300.1 · Le Tre Regole del Do-Calculus

La sfida pratica dell'inferenza causale è che nelle applicazioni reali si dispone di dati osservazionali (o di un numero limitato di esperimenti), ma si vogliono stimare quantità interventistiche $P(\mathbf{Y} \mid do(\mathbf{X} = \mathbf{x}))$ o controfattuali. Il **do-calculus** (Pearl 1995) è un sistema di tre regole che permette di manipolare espressioni miste contenenti $do()$ e di trasformarle in espressioni puramente osservazionali quando l'identificabilità è possibile, o di dimostrare che non è possibile quando non lo è:

Teorema — Le Tre Regole del Do-Calculus (Pearl 1995)

Siano $\mathbf{X}, \mathbf{Y}, \mathbf{Z}, \mathbf{W}$ insiemi disgiunti di variabili e $\mathcal{G}$ il DAG causale. La notazione $\mathcal{G}_{\overline{\mathbf{X}}}$ denota il grafo con gli archi entranti in $\mathbf{X}$ rimossi; $\mathcal{G}_{\underline{\mathbf{X}}}$ con gli archi uscenti da $\mathbf{X}$ rimossi.

Regola 1 (Inserzione/cancellazione di osservazioni): $P(\mathbf{y} \mid do(\mathbf{x}), \mathbf{z}, \mathbf{w}) = P(\mathbf{y} \mid do(\mathbf{x}), \mathbf{w})$ se $(\mathbf{Y} \perp\!\!\!\perp \mathbf{Z} \mid \mathbf{X}, \mathbf{W})_{\mathcal{G}_{\overline{\mathbf{X}}}}$.

Regola 2 (Scambio azione/osservazione): $P(\mathbf{y} \mid do(\mathbf{x}), do(\mathbf{z}), \mathbf{w}) = P(\mathbf{y} \mid do(\mathbf{x}), \mathbf{z}, \mathbf{w})$ se $(\mathbf{Y} \perp\!\!\!\perp \mathbf{Z} \mid \mathbf{X}, \mathbf{W})_{\mathcal{G}_{\overline{\mathbf{X}}\underline{\mathbf{Z}}}}$.

Regola 3 (Inserzione/cancellazione di azioni): $P(\mathbf{y} \mid do(\mathbf{x}), do(\mathbf{z}), \mathbf{w}) = P(\mathbf{y} \mid do(\mathbf{x}), \mathbf{w})$ se $(\mathbf{Y} \perp\!\!\!\perp \mathbf{Z} \mid \mathbf{X}, \mathbf{W})_{\mathcal{G}_{\overline{\mathbf{X}}\overline{\mathbf{Z(W)}}}}$ dove $\mathbf{Z(W)}$ denota le variabili di $\mathbf{Z}$ non antenate di $\mathbf{W}$ in $\mathcal{G}_{\overline{\mathbf{X}}}$.

Shpitser e Pearl (2006) hanno dimostrato che il do-calculus è **completo**: un effetto causale è identificabile se e solo se il do-calculus può ridurlo a una quantità osservazionale. Il loro algoritmo ID fornisce questa riduzione esplicitamente quando possibile e certifica la non-identificabilità altrimenti, risolvendo definitivamente il problema dell'identificabilità causale per i DAG senza archi bidirezionali tra variabili osservate.

300.2 · Il Criterio Backdoor e la Formula di Aggiustamento

Il **criterio backdoor** è la condizione sufficiente più semplice e usata per l'identificabilità: un insieme $\mathbf{Z}$ soddisfa il criterio backdoor per (X, Y) in $\mathcal{G}$ se (i) nessun nodo di $\mathbf{Z}$ è discendente di X e (ii) $\mathbf{Z}$ blocca tutti i percorsi backdoor (con arco entrante in X) tra X e Y. In tal caso: $P(Y = y \mid do(X = x)) = \sum_{\mathbf{z}} P(Y = y \mid X = x, \mathbf{Z} = \mathbf{z}) P(\mathbf{Z} = \mathbf{z})$. Questa formula di aggiustamento per covariate è la generalizzazione formale della standardizzazione epidemiologica e della regressione

multipla: si stratifica la distribuzione di Y per i valori di X e $\mathbf{Z}$, poi si media sui valori di $\mathbf{Z}$ secondo la distribuzione marginale — eliminando così l'effetto confondente di $\mathbf{Z}$ sulla relazione $X \to Y$. Il **criterio frontdoor**, applicabile quando i confondenti sono latenti ma esiste un mediatore M senza confonditori, fornisce la formula $P(Y \mid do(X = x)) = \sum_m P(M = m \mid X = x) \sum_{x'} P(Y \mid X = x', M = m) P(X = x')$ — un'identità non ovvia che permette di identificare l'effetto causale di X su Y senza osservare i confondenti, purché si osservi il mediatore.

La Gerarchia PCH: Associazione, Intervento e Controfattuale

I tre livelli della Ladder of Causation — cosa può imparare un sistema AI da dati osservazionali, interventistici e controfattuali.

301.1 · Tre Livelli di Sofisticazione Causale

Pearl e Mackenzie (2018) organizzano le interrogazioni causali in tre livelli gerarchici — la **Pearl Causal Hierarchy (PCH)** o "Ladder of Causation" — che corrispondono a capacità cognitive crescenti e richiedono tipi di dati e modelli progressivamente più ricchi. Il risultato fondamentale è che nessun algoritmo operante al livello i può rispondere a interrogazioni del livello $j > i$ senza assunzioni aggiuntive strutturali — un teorema di impossibilità che delimita i confini epistemici del machine learning puramente osservazionale.

Livello	Verbo	Interrogazione tipica
1 — Associazione	Vedere, osservare	$P(Y \mid X = x)$ — "Cosa osservo per Y se vedo $X = x$?"
2 — Intervento	Fare, intervenire	$P(Y \mid do(X = x))$ — "Cosa succede a Y se forzo $X =$
3 — Controfattuale	Immaginare, riflettere	$P(Y_x = y \mid X = x', Y = y')$ — "Cosa sarebbe successo

Il livello 1 è il dominio del machine learning standard: regressione, classificazione, stima di densità. Qualunque modello che massimizza la log-likelihood osservazionale opera in questo livello. Il livello 2 richiede informazioni causali esogene — tipicamente un DAG causale identificato da assunzioni di dominio o da esperimenti randomizzati — e permette di calcolare gli effetti delle politiche di intervento prima di implementarle. Il livello 3 richiede il modello strutturale completo con le funzioni f_i e la distribuzione sulle esogene $P(\mathbf{U})$: esso risponde a domande retrospettive che richiedono di "far girare il mondo in modo diverso" mantenendo fissi i fattori idiosincratici individuali. La medicina personalizzata ("Questo paziente si sarebbe ripreso senza il farmaco?"), la responsabilità legale ("La negligenza ha causato il danno?") e la giustizia

algoritmica (§306) richiedono il livello 3.

Identificabilità Causale e l'Algoritmo IDA

Quando è possibile stimare effetti causali da dati osservazionali — criteri grafici, CPDAG, bounds sull'effetto causale.

302.1 · Il Problema dell'Identificabilità Causale

L'identificabilità causale è il problema di determinare se e come una quantità causale $P(\mathbf{Y} \mid do(\mathbf{X}))$ possa essere espressa in termini di distribuzioni osservazionali $P(\mathbf{V})$. Il criterion backdoor e frontdoor forniscono condizioni sufficienti per l'identificabilità; il do-calculus è completo per la classe dei DAG senza variabili latenti. La difficoltà pratica è che nelle applicazioni reali il DAG causale non è noto con certezza: è stato stimato da dati osservazionali tramite causal discovery (§303-304), e questa stima è soggetta a incertezza nella struttura del grafo.

L'algoritmo **IDA (Intervention calculus when the DAG is Absent)** (Maathuis, Kalisch e Bühlmann, JASA 2009) affronta questa difficoltà: invece di un singolo DAG, si lavora con la **Markov Equivalence Class** di DAG compatibili con i dati osservazionali, rappresentata da un CPDAG (Complete Partially Directed Acyclic Graph). Per ogni DAG nell'equivalence class, si calcola l'effetto causale $P(Y \mid do(X))$ tramite il criterio backdoor o il do-calculus, ottenendo un insieme di possibili valori dell'effetto causale. Maathuis et al. hanno dimostrato che questo insieme è calcolabile efficientemente in tempo polinomiale nel numero di variabili, producendo un bound dell'effetto causale che è valido per qualunque DAG nell'equivalence class. L'IDA ha trovato applicazione nella genomica causale: identificare quali geni, se perturbati, causano cambiamenti nell'espressione di geni target, usando dati osservazionali di espressione genica e un CPDAG stimato dall'algoritmo PC.

302.2 · Identificabilità tramite Non-linearità: il Teorema di Peters-Mooij

Mentre il do-calculus e l'algoritmo IDA si applicano quando il DAG causale è (parzialmente) noto o stimabile tramite test di indipendenza condizionale, un'intera classe di metodi sfrutta una forma di identificabilità più sottile: la **asimmetria funzionale** tra causa ed effetto. Nel modello additivo $Y = f(X) + N$ con $N \perp X$, se si vuole stabilire se X causa Y oppure viceversa, il criterio di Jarfo-Mooij-Peters (2011) afferma che il modello causale (con N statisticamente indipendente dalla causa) è identificabile in modo univoco per quasi tutte le coppie (f, P_X) — con l'eccezione notevole del caso lineare gaussiano, dove entrambe le direzioni sono compatibili

con dati osservazionali. Il motivo dell'identificabilità è intuitivo: se Y fosse la causa, il rumore ε nel modello inverso $X = g(Y) + \varepsilon$ risulterebbe statisticamente dipendente da Y per quasi tutte le funzioni non lineari f e distribuzioni P_X — la dipendenza tra residui e causa è la firma dell'inversione sbagliata. Questo principio è alla base dei test HSIC-based per la direzione causale bivariata, che calcolano la dipendenza tra X e i residui del modello di regressione YoX usando kernel methods: se la dipendenza è bassa, XoY è la direzione causale corretta. Estensioni come ANM (Additive Noise Models), PNL (Post Non-Linear), e LiNGAM (per il caso lineare non gaussiano) hanno reso questi metodi applicabili a dataset reali, identificando relazioni causali bivariate con accuratezza superiore al 70% su benchmark biologici e fisici.

Causal Discovery: l'Algoritmo PC e il Constraint-Based Approach

Inferire la struttura causale dai dati osservazionali — faithfulness, v-structure, Meek rules e la CPDAG.

303.1 · Il Problema del Causal Discovery

Il **causal discovery** (o structure learning) è il problema di inferire la struttura del DAG causale $\mathcal{G}$ a partire da dati osservazionali $\{(\mathbf{v}_1, \ldots, \mathbf{v}_n)^{(i)}\}_{i=1}^{N}$. Il problema è fondamentalmente sotto-determinato: la stessa distribuzione osservazionale $P(\mathbf{V})$ è compatibile con un'intera classe di DAG equivalenti (la Markov equivalence class). Il teorema di Verma e Pearl (1990) caratterizza questa classe: due DAG sono Markov equivalenti se e solo se hanno lo stesso scheletro non orientato e gli stessi **v-structure** (o immoralità) — i pattern collider $X \to Z \leftarrow Y$ con X e Y non adiacenti. La Markov equivalence class è quindi identificata dall'insieme degli scheletri e delle immoralità, e rappresentata compattamente dal CPDAG dove gli archi orientati compaiono in tutti i DAG equivalenti e gli archi non orientati potrebbero andare in entrambe le direzioni.

L'algoritmo **PC** (Peter Spirtes e Clark Glymour, il nome viene dai due autori, 1991) è il metodo constraint-based fondativo per il causal discovery sotto le assunzioni di Markov e *faithfulness* (le sole indipendenze condizionali nella distribuzione sono quelle implicite dal DAG — nessuna cancellazione accidentale). L'algoritmo procede in due fasi. Nella prima, il *skeleton learning*, si inizia con il grafo completo non orientato e si rimuovono progressivamente gli archi (X, Y) per i quali esiste un insieme di separazione S_{XY} tale che $X \perp\!\!\!\perp Y \mid S_{XY}$ nei dati: si eseguono test di indipendenza condizionale di ordine crescente (da $|S| = 0$ a $|S| = k_{\max}$), usando χ^2, Fisher-Z, o kernel-based tests. Nella seconda fase, l'*orientamento*, si orientano le immoralità $X - Z - Y$

come $X \to Z \leftarrow Y$ dove $Z \notin S_{XY}$, e si propagano gli orientamenti usando le quattro regole di Meek (1995) per evitare cicli e nuove immoralità. Il risultato è il CPDAG che rappresenta l'intera Markov equivalence class compatibile con i dati.

FCI: Causal Discovery con Variabili Latenti

Oltre la causal sufficiency — grafi PAG, archi bidirezionali e l'inferenza causale in presenza di confondenti nascosti.

304.1 · Il Limite della Causal Sufficiency e l'Algoritmo FCI

L'algoritmo PC assume la **causal sufficiency**: tutte le cause comuni tra le variabili osservate sono esse stesse osservate. Questa assunzione è quasi sempre violata nelle applicazioni reali: in epidemiologia, i fattori genetici e socioeconomici confondono le relazioni tra esposizioni e outcomes; in econometria, variabili di policy non osservate confondono le relazioni tra variabili macroeconomiche; in neuroscienze, le oscillazioni di campo locale confondono le correlazioni tra attività di diverse regioni. L'algoritmo **FCI (Fast Causal Inference)** (Spirtes, Meek e Richardson 1995, versione efficiente di Zhang 2008) estende il PC al caso di possibili variabili latenti e produce un **PAG (Partial Ancestral Graph)**, un grafo con tre tipi di endpoint per gli archi: freccia ($\to$), cerchio ($\circ$, che indica incertezza), e coda ($-$).

Nel PAG, l'arco $X \circ\!\!\to Y$ indica che Y non è un antenato di X nel grafo causale (c'è una freccia verso Y), ma il tipo di endpoint a sinistra è incerto: X potrebbe essere un antenato di Y (freccia certa verso Y, $X -\!\!> Y$) oppure ci potrebbe essere un confondente latente tra X e Y (arco bidirezionale $X \leftrightarrow Y$). L'arco bidirezionale $X \leftrightarrow Y$ nel PAG indica la presenza certa di un confondente latente — una variabile L non osservata con archi $L \to X$ e $L \to Y$. L'algoritmo FCI è corretto e completo per la Markov equivalence class sotto variabili latenti: il PAG prodotto rappresenta esattamente l'insieme di tutti i DAG (con variabili latenti) compatibili con le indipendenze condizionali osservate nei dati, non perdendo nessuna informazione sul grafo causale recuperabile dai dati osservazionali soli.

Causal Reinforcement Learning

Agenti che apprendono e sfruttano la struttura causale dell'ambiente — robustezza, transfer e counterfactual planning.

305.1 · Correlazioni Spurie nel RL Standard

Un agente di reinforcement learning standard impara una policy $\pi(a \mid s)$ che massimizza la reward cumulativa attesa attraverso l'interazione con l'ambiente. Senza una rappresentazione esplicita delle relazioni causali tra azioni, stati e reward, l'agente è vulnerabile a due patologie fondamentali. La prima è lo sfruttamento di *correlazioni spurie*: l'agente impara ad agire in base a segnali correlati con la reward nel training environment ma non causalmente connessi ad essa — quando l'ambiente cambia, le correlazioni spurie si rompono e la policy fallisce. La seconda è la *sample inefficiency*: senza capire le cause degli outcomes, l'agente non può trasferire la conoscenza tra ambienti con la stessa struttura causale ma parametri diversi, dovendo ri-imparare da zero.

Il **Causal RL** integra gli SCM di Pearl nel ciclo di apprendimento. I principali approcci includono: (1) l'apprendimento esplicito di un *causal world model* — uno SCM che descrive le relazioni causali tra variabili di stato, azioni e reward — che permette il ragionamento controfattuale sulle conseguenze di azioni non ancora intraprese; (2) l'uso del do-calculus per stimare $P(R \mid do(a))$ tramite dati osservazionali off-policy, aggirando la necessità di esperienza diretta per ogni azione; (3) il principio di *causal invariance* che seleziona politiche il cui comportamento causale rimane invariante attraverso ambienti con la stessa struttura causale (ispirato a IRM — Invariant Risk Minimization, §216 del Volume IV). Questi approcci hanno dimostrato miglioramenti significativi nella sample efficiency e nella robustezza agli shift distributivi rispetto al RL standard, in particolare per tasks di manipolazione robotica dove la struttura causale oggetto-effetto è identificabile dalla geometria della scena.

Counterfactual Fairness: Equità Individuale attraverso la Causalità

Definizioni causali di fairness algoritmica — indipendenza controfattuale, path-specific effects e l'impossibilità delle definizioni statistiche.

306.1 · Il Fallimento delle Definizioni Statistiche di Fairness

Le definizioni statistiche di fairness — *demographic parity* $P(\hat{Y} = 1 \mid A = 0) = P(\hat{Y} = 1 \mid A = 1)$, *equalized odds*, *calibration* — operano al livello 1 della PCH e soffrono di tre limitazioni strutturali. Prima, sono mutuamente incompatibili: il teorema di impossibilità di Chouldechova (2017) e di Kleinberg et al. (2016) dimostrano che in presenza di base rates diverse tra gruppi, calibrazione, equalized odds e positive predictive value parity non possono essere soddisfatte

simultaneamente, salvo che il classificatore sia perfetto. Seconda, operano a livello di gruppo e non catturano l'equità individuale: una distribuzione aggregata equa può nascondere disparità profonde per sottogruppi o individui specifici. Terza, non distinguono tra la discriminazione causata direttamente dall'attributo protetto e correlazioni statistiche non causalmente connesse ad esso — il problema del "proxy discrimination" dove attributi apparentemente neutri fungono da proxy per l'attributo protetto.

La **Counterfactual Fairness** (Kusner, Loftus, Russell e Silva, NeurIPS 2017) definisce la fairness al livello 3 della PCH: un predittore $\hat{Y}$ è individualmente equo rispetto all'attributo protetto A se, per ogni individuo con osservazioni $(A = a, \mathbf{X} = \mathbf{x})$, la predizione è la stessa che avrebbe avuto se l'individuo fosse appartenuto a un gruppo demografico diverso, *mantenendo fissi tutti gli altri fattori causalmente indipendenti da* A:

$$P\Big(\hat{Y}_{A\leftarrow a}(\mathbf{U}) = y \mid \mathbf{X} = \mathbf{x}, A = a\Big) = P\Big(\hat{Y}_{A\leftarrow a'}(\mathbf{U}) = y \mid \mathbf{X} = \mathbf{x}, A = a\Big) \quad \forall a, a', y$$

Per costruire un predittore counterfactually fair si identifica, tramite il DAG causale, quali feature $\mathbf{X}$ sono discendenti causali di A (effetti che non devono essere usati dal predittore, pena discriminazione diretta) e quali sono causalmente indipendenti da A (possono essere usate senza violare la fairness causale). In pratica si ottiene costruendo predittori sulle variabili esogene $\mathbf{U}$ — le cause comuni che non passano attraverso A — o sulle residue della regressione di ogni feature su A, ottenendo predizioni invarianti al cambio controfattuale di A.

Transportability: Trasferire le Stime Causali tra Popolazioni

Validità esterna dei risultati sperimentali — selection diagrams, do-calculus esteso e l'algoritmo di trasporto.

307.1 · Il Problema della Validità Esterna

Uno studio sperimentale randomizzato condotto su una popolazione specifica (i partecipanti a una trial clinica, gli utenti di una piattaforma digitale durante un periodo specifico) stima $P^*(Y \mid do(X = x))$ nella popolazione sorgente Π. Applicare questo risultato alla popolazione target Π^* — pazienti con diverse caratteristiche demografiche, utenti in un diverso contesto culturale — non è automaticamente giustificato: se le distribuzioni delle variabili rilevanti

differiscono tra Π e Π^*, la stima potrebbe essere fortemente distorta. La **Transportability Theory** (Pearl e Bareinboim, 2011-2014) fornisce un framework formale per rispondere alla domanda: quando e come è legittimo trasferire una stima causale da Π a Π^*, e quale formula di trasporto correggere per le differenze tra le due popolazioni?

Il framework usa i **selection diagrams**: DAG aumentati con nodi speciali S_i (variabili di selezione, rappresentati con quadrati) che rappresentano le differenze tra le due popolazioni — un arco $S_i \to V_i$ indica che il meccanismo che genera V_i differisce tra Π e Π^*. Una stima causale $P(Y \mid do(X))$ è *transportable* da Π a Π^* se esiste una derivazione nel do-calculus esteso (con la semantica delle variabili S) che esprime $P^*(Y \mid do(X))$ in termini di distribuzioni osservabili in Π^* e quantità sperimentali da Π. L'algoritmo TRansport$^+$ (Bareinboim e Pearl 2013) è completo: determina se la stima è trasportabile, e se sì, produce la formula di trasporto esplicita. La formula di trasporto generalizza la standardizzazione diretta e indiretta dell'epidemiologia classica, permettendo correzioni per le differenze nelle distribuzioni delle covariate, nelle politiche di selezione e nei meccanismi causali stessi.

Varietà Riemanniane: Tensore Metrico, Geodetiche e Curvatura

Il linguaggio geometrico per reti su spazi curvi — simboli di Christoffel, exponential map e trasporto parallelo.

308.1 · La Struttura Riemanniana come Generalizzazione degli Spazi Euclidei

Molti tipi di dati rilevanti per il machine learning moderno vivono naturalmente su spazi curvi: le proteine si trasformano per rotazione e traslazione nello spazio fisico (§311), i grafi di conoscenza hanno strutture gerarchiche che si embeddinano in spazi iperbolici (§309), le mesh di superfici 3D hanno geometria intrinseca governata dall'operatore di Laplace-Beltrami (§312). Il formalismo delle **varietà riemanniane** fornisce il linguaggio matematico unificante per tutti questi contesti.

Una **varietà riemanniana** $(\mathcal{M}, g)$ è uno spazio topologico localmente omeomorfo a $\mathbb{R}^d$ dotato di un **tensore metrico** g che assegna un prodotto interno ai vettori tangenti in ogni punto: $g_p : T_p\mathcal{M} \times T_p\mathcal{M} \to \mathbb{R}$. In coordinate locali $\{x^i\}$, il tensore metrico è la matrice $g_{ij}(p) = g_p(\partial/\partial x^i, \partial/\partial x^j)$. La distanza tra due punti è la lunghezza della **geodetica** che li connette — la curva più corta $\gamma : [0,1] \to \mathcal{M}$ che soddisfa l'equazione geodetica:

$$\ddot{\gamma}^k + \Gamma^k_{ij}\dot{\gamma}^i\dot{\gamma}^j = 0, \qquad \Gamma^k_{ij} = \frac{1}{2}g^{kl}(\partial_i g_{jl} + \partial_j g_{il} - \partial_l g_{ij})$$

dove Γ^k_{ij} sono i **simboli di Christoffel**, funzioni della metrica e delle sue derivate prime. L'**exponential map** $\exp_p : T_p\mathcal{M} \to \mathcal{M}$ mappa un vettore tangente v al punto raggiunto dalla geodetica che parte da p nella direzione v con lunghezza $\|v\|_g$. La sua inversa locale, la **logarithmic map** $\log_p : \mathcal{M} \to T_p\mathcal{M}$, soddisfa $d(p,q) = \|\log_p(q)\|_g$ e fornisce la "differenza" tra due punti della varietà nello spazio tangente. Per la sfera S^d, le geodetiche sono i grandi cerchi; per lo spazio iperbolico $\mathbb{H}^d$, sono i diametri e i semicerchi ortogonali al bordo nel modello di Poincaré. Il **trasporto parallelo** $\Pi_\gamma^{s\to t}$ trasporta un vettore tangente lungo una curva mantenendone norma e angolo rispetto alle geodetiche — è l'operazione fondamentale per definire gradienti riemanniani e ottimizzazione su varietà (§314).

Hyperbolic Neural Networks: Modello di Poincaré e Möbius Addition

Lo spazio iperbolico come ambiente naturale per le gerarchie — embedding efficiente, Gyrovector e reti HNN complete.

309.1 · Il Volume Iperbolico e l'Efficienza per le Strutture ad Albero

Un albero binario di profondità n ha 2^n foglie. Embeddinarlo in $\mathbb{R}^d$ con distorsione controllata richiede $d = O(n)$ dimensioni, poiché il volume euclideo cresce polinomialmente. Lo **spazio iperbolico** $\mathbb{H}^d$ con curvatura sezionale costante $K = -1$ ha volume che cresce *esponenzialmente* con il raggio geodetico r: $\text{Vol}(B_r) \sim e^{(d-1)r}$ — esattamente come il numero di nodi a distanza r dalla radice in un albero regolare. Sarkar (2011) ha dimostrato che qualunque albero con n nodi può essere embeddato in $\mathbb{H}^2$ (solo 2 dimensioni!) con distorsione $\leq 1 + \varepsilon$ per qualunque $\varepsilon > 0$ scelto a priori, usando una costruzione geometrica elegante basata sui settori geodetici. Questo è impossibile nell'euclideo per qualunque dimensione fissa.

Nel **modello del disco di Poincaré** con curvatura $-c$, il dominio è $\mathbb{D}^n_c = \{\mathbf{x} \in \mathbb{R}^n : c\|\mathbf{x}\|^2 < 1\}$ con metrica conforme $g^{\mathbb{D}}_{\mathbf{x}} = \lambda^2_{\mathbf{x}} g^E$ dove $\lambda_{\mathbf{x}} = 2/(1 - c\|\mathbf{x}\|^2)$. La **Möbius addition** (somma gyrovettoriale) è l'operazione fondamentale che sostituisce la somma vettoriale euclidea:

$$\mathbf{x} \oplus_c \mathbf{y} = \frac{(1 + 2c\langle \mathbf{x}, \mathbf{y}\rangle + c\|\mathbf{y}\|^2)\mathbf{x} + (1 - c\|\mathbf{x}\|^2)\mathbf{y}}{1 + 2c\langle \mathbf{x}, \mathbf{y}\rangle + c^2\|\mathbf{x}\|^2\|\mathbf{y}\|^2}$$

La distanza geodetica diventa $d_c(\mathbf{x},\mathbf{y}) = \frac{2}{\sqrt{c}}\tanh^{-1}(\sqrt{c}\|-\mathbf{x}\oplus_c \mathbf{y}\|)$. Le **Hyperbolic Neural Networks (HNN)** (Ganea, Bécigneul e Hofmann, NeurIPS 2018) ridefiniscono le operazioni fondamentali delle reti neurali in termini della Möbius addition e delle mappe $\exp_p$, $\log_p$: la trasformazione lineare iperbolica diventa $f(\mathbf{x}) = \exp_{\mathbf{0}}(W\log_{\mathbf{0}}(\mathbf{x}) + \mathbf{b}_E)$, la non-linearità si applica nel tangent space, e la normalizzazione usa la distanza geodetica. Questa architettura sfrutta la geometria iperbolica non solo nell'embedding ma in tutta la computazione, ottenendo rappresentazioni di grafi gerarchici con distorsione sistematicamente inferiore rispetto alle reti euclidee a parità di dimensioni.

Reti Sferiche e la Convoluzione su SO(3)

Equivarianza sul gruppo di rotazioni — armoniche sferiche, S2CNN e applicazioni ai dati direzionali globali.

310.1 · Armoniche Sferiche e la Trasformata di Fourier su S2

La sfera S^2 è il dominio naturale per dati direzionali: orientamenti di molecole, direzioni del vento a scala globale, radiazione cosmica di fondo (CMB), immagini panoramiche a 360°. Una rete che non rispetta la simmetria rotazionale di S^2 deve imparare l'equivarianza dai dati, richiedendo enormi quantità di esempi augmentati. Le reti sferiche incorporano la simmetria per costruzione tramite la **trasformata di Fourier su** S^2: ogni funzione $f : S^2 \to \mathbb{R}$ si espande in **armoniche sferiche**:

$$f(\theta,\phi) = \sum_{l=0}^{\infty}\sum_{m=-l}^{l} \hat{f}_l^m Y_l^m(\theta,\phi), \qquad \hat{f}_l^m = \int_{S^2} f(\theta,\phi)\,\overline{Y_l^m(\theta,\phi)}\,\sin\theta\,d\theta\,d\phi$$

Le armoniche sferiche Y_l^m sono le autofunzioni dell'operatore di Laplace-Beltrami su S^2 con autovalore $-l(l+1)$, e formano una base ortonormale per $L^2(S^2)$. La convoluzione sferica $f \star_R g$ di due segnali su S^2 rispetto al gruppo di rotazioni $SO(3)$ è una moltiplicazione nello spazio spettrale. Le **S2CNN** (Cohen, Geiger, Köhler e Welling, ICLR 2018) implementano layer di convoluzione su S^2 con output su $SO(3)$ (il gruppo di rotazioni) tramite la trasformata di Fourier su $SO(3)$ (basata sui D-matrices di Wigner), ottenendo equivarianza esatta rispetto a qualunque rotazione. Applicazioni di successo includono la classificazione di segnali CMB

con invarianza ai frame di riferimento cosmologici, il riconoscimento di forme molecolari 3D, e la previsione meteorologica su griglia globale sfericale.

310.2 · La Distribuzione von Mises-Fisher e le Reti Probabilistiche su Sfera

L'analogo sferico della distribuzione gaussiana è la **von Mises-Fisher (vMF)**, che descrive dati concentrati attorno a una direzione media su S^{d-1}. La sua densità rispetto alla misura superficiale uniforme è $f(\mathbf{x}; \boldsymbol{\mu}, \kappa) = C_d(\kappa) \exp(\kappa\, \boldsymbol{\mu}^T \mathbf{x})$ dove $\boldsymbol{\mu} \in S^{d-1}$ è la direzione media, $\kappa \geq 0$ la concentrazione (la precisione sferica: $\kappa = 0$ dà l'uniforme, $\kappa o \infty$ il punto singolo $\boldsymbol{\mu}$), e $C_d(\kappa) = \kappa^{d/2-1} / ((2\pi)^{d/2} I_{d/2-1}(\kappa))$ è la costante di normalizzazione in termini della funzione di Bessel modificata I_u. La vMF gode di una proprietà di sufficienza analoga alla gaussiana: per campioni i.i.d., la somma $\sum_i \mathbf{x}_i$ è statistica sufficiente per $(\boldsymbol{\mu}, \kappa)$. Questo la rende ideale come prior e posterior in modelli bayesiani su sfera, e come distribuzione dei vettori di embedding normalizzati — pratica diffusa nei modelli linguistici e nel contrastive learning, dove le rappresentazioni $\mathbf{z} = f(\mathbf{x})/\|f(\mathbf{x})\|$ vivono su S^{d-1} per costruzione.

Le applicazioni delle reti sferiche spaziano dalla cosmologia alla chimica computazionale. Nella **predizione meteorologica globale**, i modelli come PanguWeather e GraphCast operano su griglie sferiche dove l'equivarianza rispetto alle rotazioni geografiche è una proprietà auspicabile — sebbene la rotazione rispetto all'asse polare sia l'unica simmetria fisica rilevante, e non l'intero gruppo $SO(3)$. Nella **cristallografia** e nella **caratterizzazione di materiali**, i pattern di diffrazione a raggi X sono naturalmente dati su sfera, e le reti sferiche equivarianti permettono di inferire la struttura tridimensionale del reticolo cristallino senza effettuare augmentation artificiale degli orientamenti. Nella modellazione del **microbioma intestinale**, i dati di abbondanza relativa vivono sul simplesso — geometricamente equivalente a una sfera con curvatura positiva dopo una trasformazione di coordinate — e l'invarianza alla riscalatura dei dati composizionali si traduce in equivarianza sferica.

SO(3) per Molecole: E(3)-Equivarianza, Irreps e AlphaFold2

Reti equivarianti per la struttura proteica — coefficienti di Clebsch-Gordan, e3nn e Invariant Point Attention.

311.1 · Il Principio di Equivarianza nella Modellazione Molecolare

Le proprietà fisiche delle molecole hanno simmetrie precise: l'energia è invariante rispetto a rotazioni, traslazioni e riflessioni (gruppo $E(3)$); le forze atomiche si trasformano come vettori

3D; i tensori di polarizzabilità si trasformano come tensori del secondo ordine. Una rete non equivariante che riceve le coordinate cartesiane degli atomi deve imparare queste simmetrie dai dati, richiedendo training su milioni di configurazioni diverse. Le reti equivarianti incorporano la simmetria $E(3)$ per costruzione, riducendo di ordini di grandezza la quantità di dati necessaria.

Le **rappresentazioni irriducibili (irreps)** di $SO(3)$ sono indicizzate dal grado l: $l = 0$ per scalari invarianti, $l = 1$ per vettori (trasformazione come $\mathbb{R}^3$ sotto rotazioni), $l = 2$ per tensori del secondo ordine (5 componenti indipendenti). Il prodotto tensoriale di irreps di gradi l_1 e l_2 si decompone in irreps di gradi $|l_1 - l_2| \leq l \leq l_1 + l_2$ tramite i **coefficienti di Clebsch-Gordan**: $\rho^{l_1} \otimes \rho^{l_2} = \bigoplus_{l=|l_1-l_2|}^{l_1+l_2} C^{l_1 l_2 l} \rho^l$. Le reti equivarianti come **e3nn** (Geiger e Smidt 2022) costruiscono layer che operano su feature multi-grado $\mathbf{x} = (\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots)$ e producono aggiornamenti equivarianti usando i coefficienti di Clebsch-Gordan come pesi di mixing. **AlphaFold2** usa l'**Invariant Point Attention (IPA)** che mantiene l'equivarianza rispetto al frame di riferimento locale di ogni residuo amminoacidico: gli aggiornamenti di attenzione sono calcolati in coordinate invarianti rispetto al frame locale, e i frame vengono aggiornati tramite trasformazioni rigide equivarianti — il componente architetturale chiave che permette la predizione della struttura proteica con accuratezza quasi-atomica.

Mesh Processing: Laplace-Beltrami, Cotangent Weights e GNN su Superfici

Apprendimento intrinseco su mesh 3D — discretizzazione dell'operatore di Laplace-Beltrami, MeshCNN e invarianza isometrica.

312.1 · La Discretizzazione dell'Operatore di Laplace-Beltrami

Una mesh poligonale $\mathcal{G} = (V, E, F)$ — insieme di vertici, archi e facce triangolari — è la rappresentazione standard per superfici 3D in computer grafica, robotica e biologia strutturale. L'operatore fondamentale per l'analisi geometrica su superfici è il **Laplace-Beltrami** $\Delta_{\mathcal{M}}$, che generalizza il laplaciano euclideo alle varietà riemanniane. Per una funzione $f : V \to \mathbb{R}$ definita sui vertici, la discretizzazione con i **cotangent weights** (Meyer, Desbrun, Schröder e Barr 2003) è:

$$(\Delta_{\mathcal{M}} f)_i = \frac{1}{2A_i} \sum_{j \in \mathcal{N}(i)} (\cot \alpha_{ij} + \cot \beta_{ij})(f_j - f_i)$$

dove α_{ij} e β_{ij} sono gli angoli opposti all'arco (i, j) nelle due facce adiacenti, e A_i è l'area della Voronoi cell del vertice i. I cotangent weights sono l'unica discretizzazione del Laplace-Beltrami che è *consistente* (converge all'operatore continuo al raffinamento della mesh), *hermitiana* (matrice laplaciana simmetrica) e *invariante rispetto alle isometrie* — stiramenti e piegature della mesh che preservano le distanze geodetiche non cambiano il Laplace-Beltrami discretizzato. Le reti neurali che usano questa discretizzazione come operatore di aggregazione nei GNN — come **MeshCNN** (Hanocka et al. 2019) e **CoMA** (Ranjan et al. 2018) — sono intrinsecamente robuste alle rotazioni e traslazioni rigide della mesh, e le loro feature catturano proprietà geometriche intrinseche (curvatura, geodesiche) piuttosto che coordinate cartesiane dipendenti dall'orientamento.

DiffusionNet: Apprendimento Intrinseco Invariante alla Triangolazione

Diffusione del calore su superfici — autovalori del Laplace-Beltrami, invarianza alla rimagliatura e generalizzazione cross-mesh.

313.1 · Il Problema della Sensibilità alla Triangolazione

Le reti su mesh che aggregano informazioni lungo gli archi del grafo poligonale soffrono di un problema fondamentale: due mesh che approssimano la stessa superficie geometrica ma con diverse triangolazioni (diverso numero di triangoli, diverso orientamento degli archi) producono segnali di input diversi, impedendo la generalizzazione cross-mesh. **DiffusionNet** (Sharp, Attaiki, Crane e Ovsjanikov, ACM SIGGRAPH 2022) risolve questo problema elegantemente: invece di aggregare lungo gli archi del grafo poligonale (dipendente dalla triangolazione), propaga le feature tramite il **kernel del calore della superficie**:

$$k_t(x, y) = \sum_{i=0}^{\infty} e^{-\lambda_i t}\, \phi_i(x)\, \phi_i(y)$$

dove λ_i e ϕ_i sono autovalori e autofunzioni del Laplace-Beltrami della superficie. Poiché autovalori e autofunzioni dipendono solo dalla geometria *intrinseca* (invariante isometrica) e non dalla triangolazione, la diffusione tramite il kernel del calore è completamente invariante alla rimagliatura — due mesh con diverse triangolazioni della stessa superficie producono le stesse feature. In pratica, la decomposizione spettrale viene precalcolata sui primi k autovettori (tipicamente $k = 128$), e il tempo di diffusione t per ogni layer è un parametro appreso che con-

trolla la scala spaziale della propagazione. DiffusionNet ha stabilito nuovi state-of-the-art nella corrispondenza tra superfici (surface matching), classificazione di forme 3D, e segmentazione semantica di mesh umane, superando reti specializzate con un framework significativamente più semplice.

Ottimizzazione su Varietà Riemanniane

Riemannian gradient descent, retrazioni, ottimizzazione su Stiefel e applicazioni agli embedding iperbolici.

314.1 · Il Riemannian Gradient Descent

L'ottimizzazione $\min_{\mathbf{x}\in\mathcal{M}} f(\mathbf{x})$ su una varietà riemanniana richiede di adattare il gradient descent alla struttura geometrica non piatta. Il **Riemannian Gradient Descent (RGD)** procede in tre passi: calcolare il **gradiente riemanniano** $\operatorname{grad}_{\mathcal{M}} f(\mathbf{x}) = \mathcal{P}_{T_{\mathbf{x}}\mathcal{M}}(\nabla f(\mathbf{x}))$ proiettando il gradiente euclideo sul tangent space; calcolare la direzione di discesa $\mathbf{v} = -\alpha \operatorname{grad}_{\mathcal{M}} f(\mathbf{x})$ nel tangent space; "muoversi" sulla varietà tramite una **retrazione** $R_{\mathbf{x}}(\mathbf{v})$, che approssima l'exponential map con garanzie di primo ordine: $R_{\mathbf{x}}(\mathbf{v}) = \mathbf{x} + O(\|\mathbf{v}\|^2)$. Le retrazioni comuni includono: per la sfera S^{n-1}, la normalizzazione $R_{\mathbf{x}}(\mathbf{v}) = (\mathbf{x}+\mathbf{v})/\|\mathbf{x}+\mathbf{v}\|$; per il gruppo di Stiefel $\mathrm{St}(n,k) = \{X \in \mathbb{R}^{n\times k} : X^T X = I_k\}$ (matrici con colonne ortonormali), la decomposizione QR $R_X(G) = \mathrm{qf}(X+G)$ dove qf estrae il fattore Q; per il disco di Poincaré, l'exponential map iperbolica. Il gradiente riemanniano differisce dal gradiente euclideo per il rescaling metrico: per la sfera, $\operatorname{grad}_{S^n} f = \nabla f - \langle \nabla f, \mathbf{x}\rangle \mathbf{x}$ (proiezione sul tangent space eliminando la componente radiale). Le varianti Riemannian di Adam, AdaGrad e momentum sono state implementate nella libreria Geoopt (Kochurov et al. 2020) e usate per ottimizzare embedding iperbolici, matrici di mixing ortogonali e layer con pesi su varietà specifiche.

Gyrovector Spaces: L'Algebra della Geometria Iperbolica

L'estensione degli spazi vettoriali alla curvatura negativa — gyrogroups, gyroassociativity e analogie con la relatività speciale.

315.1 · Dalla Möbius Addition alla Struttura Algebrica

Gli spazi vettoriali euclidei hanno una struttura algebrica ricca — addizione commutativa e associativa, scalatura, combinazioni lineari — che non si trasferisce automaticamente agli spazi iperbolici. La Möbius addition $\oplus_c$ è né commutativa né associativa: $\mathbf{x} \oplus_c \mathbf{y} \neq \mathbf{y} \oplus_c \mathbf{x}$ in generale, e $(\mathbf{x} \oplus_c \mathbf{y}) \oplus_c \mathbf{z} \neq \mathbf{x} \oplus_c (\mathbf{y} \oplus_c \mathbf{z})$. Tuttavia Abraham Ungar (2008) ha identificato una struttura algebrica coerente — i **gyrovector spaces** — che è l'analogo iperbolico degli spazi vettoriali, e che soddisfa proprietà generalizzate: la *gyrocommutativity* $\mathbf{x} \oplus_c \mathbf{y} = \mathrm{gyr}[\mathbf{x}, \mathbf{y}](\mathbf{y} \oplus_c \mathbf{x})$ e la *gyroassociativity* $\mathbf{x} \oplus_c (\mathbf{y} \oplus_c \mathbf{z}) = (\mathbf{x} \oplus_c \mathbf{y}) \oplus_c \mathrm{gyr}[\mathbf{x}, \mathbf{y}]\mathbf{z}$, dove $\mathrm{gyr}[\mathbf{x}, \mathbf{y}]$ è un *gyroautomorphism* — un operatore lineare isometrico che misura la "rotazione iperbolica" indotta dalla composizione di due spostamenti. Nella fisica relativistica, i gyroautomorphismi corrispondono alla *precessione di Thomas*: la composizione di due boost di Lorentz in direzioni diverse non è un boost puro ma include una rotazione spaziale — il collegamento profondo tra la geometria iperbolica e la relatività speciale.

Il formalismo gyrovettoriale permette di definire per lo spazio di Poincaré tutte le operazioni delle reti neurali standard: la **media di Fréchet** (media iperbolica che minimizza la somma dei quadrati delle distanze geodetiche), la **regressione logistica iperbolica** (iperpiani geodetici come confini decisionali), l'**attenzione iperbolica** (softmax su distanze geodetiche), e i layer di convoluzione iperbolici. La libreria Geoopt implementa tutte queste operazioni in PyTorch con supporto per la differenziazione automatica, permettendo di costruire reti completamente native nello spazio iperbolico senza dover tornare allo spazio euclideo tangente per ogni operazione — la fonte più comune di errori nelle prime implementazioni di HNN.

GP Regression: Prior su Funzioni, Posterior e Marginal Likelihood

Il modello di regressione bayesiana non parametrico completo — kernel, predizione con incertezza e automatic relevance determination.

316.1 · I Processi Gaussiani come Prior sulle Funzioni

La regressione bayesiana parametrica pone un prior sui parametri θ di un modello f_θ; la regressione non parametrica bayesiana pone un prior direttamente sulle funzioni $f : \mathcal{X} \to \mathbb{R}$. Un **Processo Gaussiano (GP)** è una distribuzione di probabilità su funzioni tale che, per qualunque insieme finito di punti $\{x_1, \ldots, x_n\} \subset \mathcal{X}$, il vettore $(f(x_1), \ldots, f(x_n))$ ha distribuzione gaussiana congiunta. Un GP è completamente specificato da una funzione media

$m(x) = \mathbb{E}[f(x)]$ (solitamente zero per semplicità) e una **funzione di covarianza (kernel)** $k(x, x') = \mathrm{Cov}[f(x), f(x')]$. Ogni kernel definisce una classe diversa di funzioni: il **kernel RBF** $k_{\mathrm{RBF}}(x, x') = \sigma^2 \exp(-\|x - x'\|^2 / 2\ell^2)$ genera funzioni infinitamente differenziabili con scala di variazione ℓ (lunghezza di scala) e varianza σ^2; il **Matérn-5/2** genera funzioni due volte differenziabili con decadimento più rapido della covarianza; il **kernel periodico** genera funzioni che si ripetono con periodo p; il **kernel lineare** $k(x, x') = \sigma^2 x^T x'$ corrisponde alla regressione lineare bayesiana. I kernel si combinano tramite somma, prodotto e composizione, permettendo di costruire prior espressivi che codificano conoscenze di dominio sulla funzione target.

316.2 · Posterior Gaussiana e Aggiornamento Bayesiano

Dati n osservazioni rumorose $\{(x_i, y_i)\}_{i=1}^n$ con $y_i = f(x_i) + \varepsilon_i$, $\varepsilon_i \sim \mathcal{N}(0, \sigma_n^2)$, la distribuzione posteriore del GP rimane gaussiana (proprietà di coniugazione gaussiana) con forma chiusa. Per un insieme di punti test $\mathbf{X}_*$, la distribuzione predittiva è:

$$\mathbf{f}_* \mid \mathbf{X}_*, \mathcal{D} \sim \mathcal{N}(\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*)$$

$$\boldsymbol{\mu}_* = K_{*n}(K_{nn} + \sigma_n^2 I)^{-1}\mathbf{y}, \qquad \boldsymbol{\Sigma}_* = K_{**} - K_{*n}(K_{nn} + \sigma_n^2 I)^{-1}K_{n*}$$

dove $K_{nn} = k(\mathbf{X}, \mathbf{X})$ è la matrice di Gram sui punti di training (costo $O(n^2)$ in memoria), $K_{*n} = k(\mathbf{X}_*, \mathbf{X})$ la cross-kernel matrix, e $K_{**} = k(\mathbf{X}_*, \mathbf{X}_*)$. La fattorizzazione di Cholesky $L = \mathrm{chol}(K_{nn} + \sigma_n^2 I)$ (costo $O(n^3)$) risolve il sistema lineare necessario per $\boldsymbol{\mu}_*$ e $\boldsymbol{\Sigma}_*$. Gli iperparametri $(\sigma^2, \ell, \sigma_n^2)$ si ottimizzano massimizzando la **log-marginal likelihood**:

$$\log P(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta}) = -\frac{1}{2}\mathbf{y}^T(K + \sigma_n^2 I)^{-1}\mathbf{y} - \frac{1}{2}\log|K + \sigma_n^2 I| - \frac{n}{2}\log 2\pi$$

Il primo termine misura la fit (maggiore per modelli che assegnano alta probabilità ai dati), il secondo penalizza la complessità del modello (più grande per kernel più flessibili — il principio di Occam bayesiano), e la loro combinazione produce l'**Automatic Relevance Determination (ARD)**: se si usa un kernel RBF con lunghezze di scala distinte ℓ_d per ogni dimensione d dell'input, l'ottimizzazione della marginal likelihood pone automaticamente $\ell_d \to \infty$ per le dimensioni irrilevanti, azzerando la loro influenza — un meccanismo di feature selection intrinseco

al GP che non richiede regolarizzazione esplicita.

GP Classification: Approssimazione di Laplace ed EP

Processi Gaussiani per classificazione binaria e multi-classe — likelihood non gaussiane, posterior approssimata e predizione calibrata.

317.1 · Likelihood Non Gaussiana e la Necessità dell'Approssimazione

Per la classificazione binaria, la likelihood naturale è $P(y_i = 1 \mid f(x_i)) = \sigma(f(x_i))$ (funzione logistica) o $\Phi(f(x_i))$ (funzione probit) — nessuna delle due è coniugata al prior GP gaussiano. La posterior $p(f \mid \mathcal{D}) \propto p(f) \prod_i p(y_i \mid f(x_i))$ non ha forma chiusa. Due approcci principali si contendono il campo. L'**approssimazione di Laplace** (Rasmussen e Williams 2006) trova il MAP f^* della posterior (tramite Newton-Raphson sulla log-posterior concava) e approssima la posterior con una gaussiana centrata nel MAP con covarianza uguale all'Hessiano negativo della log-posterior: $q(f) = \mathcal{N}(f^*, (K^{-1} + W)^{-1})$ dove $W = -\nabla^2 \log p(\mathbf{y} \mid f^*)$ è la matrice diagonale delle curvature della likelihood. L'**Expectation Propagation** (§281) approssima invece ogni fattore di likelihood $p(y_i \mid f(x_i))$ con un fattore gaussiano, tipicamente producendo approssimazioni più accurate della Laplace per likelihood fortemente non gaussiane. In entrambi i casi, la predizione su nuovi punti richiede un integrale $P(y_* = 1 \mid x_*) = \int \sigma(f)\, q(f \mid x_*, \mathcal{D})\, df$ approssimabile con quadratura di Gauss-Hermite.

317.2 · La Posterior Approssimata e il Calcolo della Predizione

Una volta ottenuta l'approssimazione gaussiana della posterior — sia tramite Laplace sia tramite EP — la predizione su un nuovo punto x_* richiede due passi. Nel primo si calcola la distribuzione marginale predittiva $q(f_* \mid x_*, \mathcal{D})$, che per entrambe le approssimazioni ha la forma gaussiana $\mathcal{N}(\mu_*, \sigma_*^2)$ con media e varianza derivabili dalla formula standard della GP regression applicata alla posterior approssimata. Nel secondo passo si integra la likelihood non lineare rispetto a questa gaussiana, calcolando $P(y_* = 1 \mid x_*) = \int \sigma(f_*)\, q(f_*)\, df_*$: questo integrale non ha forma chiusa per la funzione logistica, ma è approssimabile con quadratura di Gauss-Hermite a cinque o dieci punti con errore trascurabile in pratica. Per la funzione probit $\Phi(f_*)$, invece, l'integrale ha una forma chiusa esatta: $P(y_* = 1 \mid x_*) = \Phi(\mu_* / \sqrt{1 + \sigma_*^2})$ — una proprietà analitica che rende il modello probit-GP particolarmente comodo in contesti dove la velocità di calcolo è critica.

La scelta tra l'approssimazione di Laplace e l'EP è guidata da considerazioni di accuratezza

e costo computazionale. Laplace è concettualmente semplice, richiede una sola passata di Newton-Raphson, e produce approssimazioni ragionevoli per likelihood non troppo lontane dalla gaussiana. EP è iterativa, più costosa per iterazione, ma cattura meglio le asimmetrie e le code pesanti della posterior — si stima empiricamente che EP riduca l'errore di calibrazione della probabilità predittiva di un fattore due rispetto a Laplace su problemi di classificazione binaria tipici. La differenza tra i due metodi è particolarmente marcata quando le classi sono molto sbilanciate o quando i dati di training sono sparsi: in questi casi, la posterior della funzione latente è fortemente non gaussiana e l'approssimazione di Laplace, che cattura solo il curvatura locale attorno al MAP, non riesce a rappresentare le code della distribuzione.

317.3 · Estensione alla Classificazione Multi-Classe

La classificazione multi-classe con $K > 2$ classi richiede K funzioni latenti $f_1, \ldots, f_K$ — una per classe — con prior GP condiviso o indipendente. La likelihood standard è la softmax: $P(y = k \mid \mathbf{f}) = e^{f_k} / \sum_j e^{f_j}$. La posterior congiunta $p(\mathbf{f}_1, \ldots, \mathbf{f}_K \mid \mathcal{D})$ è intrattabile per $K > 2$ anche con le approssimazioni standard, poiché la dipendenza tra le K funzioni latenti attraverso la softmax rende impossibile la fattorizzazione. L'approccio più usato è la variational inference con K GP indipendenti come variational family — il metodo SVGP (§320) si estende naturalmente al multi-classe — oppure la riduzione a K problemi binari di classificazione indipendenti tramite One-vs-Rest, a spese di un calibrazione meno accurata delle probabilità aggregate.

Kernel Trick e il Teorema di Mercer

Feature map implicite nello spazio di Hilbert — RKHS, Representer Theorem e la connessione tra GP e SVM.

318.1 · Kernels come Prodotti Interni in Dimensione Infinita

Il **kernel trick** è la chiave che unifica GP, SVM e metodi a kernel: invece di mappare esplicitamente i dati in uno spazio di feature ad alta (o infinita) dimensione $\phi : \mathcal{X} \to \mathcal{H}$ e computare $\langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$, si valuta direttamente $k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$ in $O(d)$ invece di $O(|\mathcal{H}|)$. Il **Teorema di Mercer** (1909) caratterizza le funzioni che sono kernel validi: $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ è positivo semi-definito se e solo se per ogni insieme finito di punti la matrice di Gram $K_{ij} = k(x_i, x_j)$ è PSD, se e solo se esiste uno spazio di Hilbert $\mathcal{H}_k$ (il **Reproducing Kernel Hilbert Space, RKHS**) con feature map ϕ tale che $k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}_k}$. Il **Representer Theorem** (Schölkopf, Herbrich e Smola 2001) afferma che il minimizzatore di qualunque funzionale

$\|f\|^2_{\mathcal{H}_k} + \sum_i L(y_i, f(x_i))$ appartiene all'RKHS e ha la forma $f^*(\cdot) = \sum_i \alpha_i k(x_i, \cdot)$ — una combinazione lineare dei kernel centrati nei punti di training. Questo giustifica sia la regressione GP (la posterior è una combinazione lineare di kernel) che il duale SVM (§328). Per il kernel RBF, la feature map è infinito-dimensionale e corrisponde a una rete neurale con infinite hidden units gaussiane — la connessione profonda tra deep learning e metodi kernel.

Sparse Gaussian Processes: Inducing Points e FITC

Scalabilità dei GP a grandi dataset — approssimazioni a basso rango, ottimizzazione degli inducing points e costo $O(nm^2)$.

319.1 · Il Collo di Bottiglia Cubico e l'Approccio Sparse

I GP esatti hanno costo $O(n^3)$ per la fattorizzazione di Cholesky e $O(n^2)$ in memoria — intrattabile per $n > 10^4$. Gli **Sparse GP** introducono $m \ll n$ **inducing points** $\mathbf{Z} = \{z_1, \ldots, z_m\} \subset \mathcal{X}$ e approssimano il GP tramite la distribuzione condizionale ai valori $\mathbf{u} = f(\mathbf{Z})$ del GP negli inducing points. L'approccio **FITC (Fully Independent Training Conditional)** (Snelson e Ghahramani 2006) approssima la prior condizionale come: $p(\mathbf{f} \mid \mathbf{u}) \approx \prod_i p(f_i \mid \mathbf{u})$ — i valori di training sono condizionalmente indipendenti dati gli inducing points. Questo porta alla matrice di covarianza approssimata:

$$K_{nn}^{\text{FITC}} = Q_{nn} + \text{diag}(K_{nn} - Q_{nn}), \quad Q_{nn} = K_{nm} K_{mm}^{-1} K_{mn}$$

dove Q_{nn} è l'approssimazione di basso rango (rango m) e il termine diagonale corregge la varianza individuale degli errori di approssimazione. Il costo si riduce a $O(nm^2)$ — praticabile per milioni di punti con $m \sim$ 500-1000 inducing points. La posizione $\mathbf{Z}$ degli inducing points viene ottimizzata congiuntamente agli iperparametri del kernel massimizzando la marginal likelihood approssimata: gli inducing points tendono a concentrarsi nelle regioni di alta densità informativa dei dati, massimizzando l'approssimazione a basso rango della matrice di covarianza.

319.2 · Il Bound Variazionale di Titsias: Fondamenta Bayesiane degli Sparse GP

Parallelamente all'approccio frequentista di FITC, Titsias (2009) ha derivato un'alternativa pienamente bayesiana che offre garanzie di convergenza più solide. Invece di approssimare la

prior del GP con una versione di basso rango, si ricava direttamente un lower bound variazionale alla vera log-marginal likelihood:

$$\log P(\mathbf{y} \mid \mathbf{X}) \geq \log \mathcal{N}(\mathbf{y} \mid K_{nm}K_{mm}^{-1}\mathbf{m}_u, \sigma_n^2 I) - \frac{1}{2\sigma_n^2}\mathrm{tr}(K_{nn} - Q_{nn})$$

dove il secondo termine è una penalità di traccia che misura quanto l'approssimazione a basso rango $Q_{nn} = K_{nm}K_{mm}^{-1}K_{mn}$ sottostima la varianza delle osservazioni. Questa traccia è sempre non negativa — una conseguenza diretta della disuguaglianza di Schur sul complemento di Schur della matrice di covarianza — il che garantisce che il bound sia veramente un lower bound e non un'approssimazione con errore incontrollato. Il vantaggio fondamentale rispetto a FITC risiede in una proprietà di completezza: quando le locazioni $\mathbf{Z}$ degli inducing points coincidono esattamente con tutti gli n punti di training, la traccia di penalità si annulla identicamente e si recupera il GP esatto. Questa garanzia di convergenza, che FITC non possiede, rende il bound di Titsias lo strumento di scelta per la selezione automatica del numero e della posizione degli inducing points tramite ottimizzazione del bound stesso — un meccanismo di parsimonia che seleziona il modello sparso più semplice compatibile con i dati.

SVGP: Stochastic Variational Gaussian Processes

Inferenza variazionale scalabile per GP — ELBO con mini-batch, deep GP e la connessione con il deep learning.

320.1 · Variational Inference per GP Sparsi

Gli **Stochastic Variational GP (SVGP)** (Hensman, Fusi e Lawrence, UAI 2013; Hensman, Matthews e Ghahramani, AISTATS 2015) combinano sparse GP e inferenza variazionale in un framework scalabile compatibile con il mini-batch training. Si introduce una distribuzione variazionale $q(\mathbf{u}) = \mathcal{N}(\mathbf{m}, \mathbf{S})$ sugli inducing values, e si ottimizza l'ELBO:

$$\mathcal{L}_{\text{SVGP}} = \sum_{i=1}^{n} \mathbb{E}_{q(f(x_i))}[\log p(y_i \mid f(x_i))] - \mathrm{KL}(q(\mathbf{u}) \| p(\mathbf{u}))$$

Il primo termine (somma sulla likelihood) è additivo sulle osservazioni e può essere stimato con mini-batch di dimensione b: il contributo di ogni osservazione è $\mathbb{E}_{q(f(x_i))}[\log p(y_i \mid f(x_i))]$,

calcolabile in $O(m^2)$ per punto data la gaussiana marginale $q(f(x_i)) = \mathcal{N}(\tilde{\mu}_i, \tilde{\sigma}_i^2)$ con $\tilde{\mu}_i = K_{im}K_{mm}^{-1}\mathbf{m}$ e $\tilde{\sigma}_i^2 = k(x_i,x_i) - K_{im}K_{mm}^{-1}K_{mi} + K_{im}K_{mm}^{-1}\mathbf{S}K_{mm}^{-1}K_{mi}$. Il secondo termine (KL tra gaussiane) ha forma chiusa. Il costo totale per iterazione con mini-batch di dimensione b è $O(bm^2 + m^3)$ — scalabile a dataset con milioni di osservazioni. SVGP è il backbone di GPflow (De G. Matthews et al. 2017) e ha abilitato i **deep GP**: stack di GP con variabili latenti intermedie addestrati con SVGP a ogni layer, analoghi alle reti profonde ma con inferenza bayesiana esatta per ogni layer.

Bayesian Optimization: UCB, EI e il Ciclo Surrogate-Acquisizione

Ottimizzazione global di funzioni costose — il GP come surrogate, acquisition functions e regret bayesiano.

321.1 · Il Framework della Bayesian Optimization

La **Bayesian Optimization (BO)** affronta il problema di ottimizzare $f^* = \max_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x})$ quando f è una *black-box* costosa da valutare: training di un modello ML, esperimento chimico, simulazione CFD. Non si dispone del gradiente, e ogni valutazione costa minuti o ore. Il ciclo BO alterna tre operazioni: (1) aggiornare un **surrogate model GP** con tutte le valutazioni effettuate finora, ottenendo $\mu(\mathbf{x})$ e $\sigma(\mathbf{x})$ per ogni punto; (2) ottimizzare analyticamente un' **acquisition function** sul surrogate per scegliere il prossimo punto da valutare $\mathbf{x}_{\text{next}} = \arg\max_{\mathbf{x}} a(\mathbf{x})$; (3) valutare $f(\mathbf{x}_{\text{next}})$ e aggiornare il GP.

Le Tre Acquisition Functions Principali

UCB (Upper Confidence Bound): $a_{\text{UCB}}(\mathbf{x}) = \mu(\mathbf{x}) + \kappa\sigma(\mathbf{x})$. Il parametro $\kappa > 0$ controlla il bilancio exploitation-exploration. Con $\kappa = \sqrt{2\log(t^{d/2+2}\pi^2/3\delta)}$ (dipendente da t e dalla dimensione d), si ottengono garanzie di regret cumulativo sublineare: $\sum_{t=1}^{T}[f(\mathbf{x}^*) - f(\mathbf{x}_t)] = O(\sqrt{T\,\gamma_T \log T})$ dove γ_T è l'*information gain* massimo su T punti.

EI (Expected Improvement): $a_{\text{EI}}(\mathbf{x}) = \mathbb{E}[\max(f(\mathbf{x}) - f^+, 0)]$ con $f^+ = \max_i y_i$. In forma chiusa per GP: $a_{\text{EI}} = (\mu - f^+)\Phi(z) + \sigma\phi(z)$ con $z = (\mu - f^+)/\sigma$. EI è exploration-aware: nelle regioni con alta incertezza anche valori medi bassi portano a EI positivo grazie alla probabilità di improvement.

Thompson Sampling: campionare $\tilde{f} \sim p(f \mid \mathcal{D})$ e scegliere $\mathbf{x}_{\text{next}} = \arg\max \tilde{f}(\mathbf{x})$. La randomizzazione garantisce la diversificazione automatica nel setting batch (§322).

Thompson Sampling e Bayesian Optimization Multi-Fidelity

Campionamento dalla posterior GP come acquisition — batch BO, valutazioni a costo variabile e speedup multi-fidelity.

322.1 · Thompson Sampling per la BO

Il **Thompson Sampling (TS)** per la Bayesian Optimization (Russo e Van Roy 2014, applicazione alla BO continua) è concettualmente il più semplice tra gli approcci: ad ogni iterazione si campiona una funzione $\tilde{f}(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x})^T \tilde{\mathbf{w}}$ con $\tilde{\mathbf{w}} \sim P(\mathbf{w} \mid \mathcal{D})$ (o equivalentemente, si campiona dal GP posterior tramite la sua decomposizione spettrale), e si valuta f nel massimizzatore di $\tilde{f}$. La randomizzazione del campionamento produce diversificazione naturale: in diverse iterazioni vengono esplorate regioni diverse dello spazio, bilanciando automaticamente l'exploitation del massimo attuale con l'exploration delle regioni incerte. TS ha garanzie di regret bayesiano ottimali, è facilmente estendibile al setting *batch* (selezionare $b > 1$ punti in parallelo campionando b funzioni indipendenti dal GP posterior), ed è empiricamente competitivo o superiore a UCB e EI nella maggior parte dei benchmark — la semplicità concettuale non implica inferiorità pratica.

322.2 · Multi-Fidelity Bayesian Optimization

In molte applicazioni pratiche, la funzione obiettivo può essere valutata a diverse *fedeltà* con costi diversi: per l'ottimizzazione degli iperparametri di una rete, si può valutare la performance addestrando su 1k, 10k o 100k esempi — con costi nell'ordine di secondi, minuti e ore rispettivamente. Il **multi-fidelity BO** (algoritmi chiave: MF-GP-UCB, MFBO, BOCA) modella la funzione obiettivo come un GP su $\mathcal{X} \times \{1, \ldots, s_{\max}\}$ con kernel di correlazione tra fedeltà: $k((x,s),(x',s')) = k_{\mathcal{X}}(x,x') \cdot \rho(s,s')$. L'acquisition function include il costo c_s di ogni fedeltà nel denominatore, ottimizzando il rapporto improvement-per-unità-di-costo. Lo speedup rispetto alla BO a singola fedeltà è tipicamente di un ordine di grandezza quando la correlazione tra le fedeltà è alta ($\rho > 0.9$) — il modello impara che le valutazioni a bassa fedeltà sono proxy affidabili per quelle costose, concentrando le valutazioni costose solo nelle regioni più promettenti.

Neural Architecture Search Bayesiano

BO per lo spazio discreto delle architetture neurali — kernel su grafi, BANANAS e DARTS differenziabile.

323.1 · Il NAS come Problema di Ottimizzazione Black-Box

Il **Neural Architecture Search (NAS)** automatizza il design delle architetture neurali: dato uno spazio di ricerca (tipi di operazione, connettività, dimensioni dei layer), si cerca l'architettura con le migliori performance sul task target. Lo spazio di ricerca tipico contiene 10^{18}-10^{30} architetture possibili — una foresta inesplorata per qualunque metodo di ricerca esaustiva. La **BO per NAS** usa un GP con kernel su grafi di architetture come surrogate model: il kernel di edit-distance $k(A, A') = \exp(-d_{\text{edit}}(A, A'))$ o il Weisfeiler-Lehman graph kernel misura la similarità strutturale tra architetture, permettendo al GP di generalizzare le performance osservate a architetture non ancora valutate. L'algoritmo **BANANAS** (White, Neiswanger e Savani 2021) usa un ensemble di reti neurali come surrogate al posto del GP, con quantificazione dell'incertezza tramite MC Dropout — più scalabile per spazi di ricerca molto grandi. Il **DARTS (Differentiable Architecture Search)** (Liu, Simonyan e Yang 2019) aggira interamente la BO rilassando il problema discreto: i pesi di mescolamento $\alpha_o^{(i,j)}$ per ciascuna operazione o sull'arco (i, j) del DAG di architettura vengono ottimizzati con gradient descent bilevel rispetto a pesi di architettura e pesi della rete congiuntamente. La derivabilità rispetto alla struttura è ottenuta tramite il Gumbel-softmax trick, che approssima l'argmax discreto con una funzione differenziabile parametrizzata dalla temperatura.

AdaBoost: L'Algoritmo e la Prova di Convergenza Esponenziale

Da learner deboli a classificatori forti — pesi adattativi, edge teorico e il bound esponenziale sull'errore di training.

324.1 · La Domanda di Kearns e Valiant

Nel 1988, Kearns e Valiant posero una domanda apparentemente accademica: è computazionalmente possibile amplificare la precisione di un learner debole (accuratezza appena superiore al 50%) fino a ottenere un classificatore arbitrariamente accurato? Freund e Schapire (1997) risposero affermativamente con l'algoritmo **AdaBoost** (Adaptive Boosting), uno dei contributi più influenti del machine learning degli anni '90 che ha vinto il Gödel Prize nel 2003. L'idea è iterativa e adattiva: a ogni round t, si addestra un *weak learner* $h_t : \mathcal{X} \to \{-1, +1\}$ su una

distribuzione pesata dei dati dove gli esempi mal classificati dai round precedenti ricevono peso maggiore; il classificatore finale è la combinazione lineare pesata dei learner:

$$H(\mathbf{x}) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})\right), \qquad \alpha_t = \frac{1}{2}\log\frac{1-\varepsilon_t}{\varepsilon_t}$$

dove $\varepsilon_t = \sum_i w_i^{(t)} \mathbf{1}[h_t(x_i) \neq y_i]$ è l'errore pesato del weak learner al round t e $\alpha_t > 0$ richiede $\varepsilon_t < 1/2$ (ogni learner deve battere il caso). I pesi si aggiornano esponenzialmente: $w_i^{(t+1)} \propto w_i^{(t)} \exp(-y_i \alpha_t h_t(x_i))$ — gli esempi classificati correttamente vedono il loro peso ridotto, quelli sbagliati amplificato, focalizzando ogni nuovo learner sugli esempi difficili.

324.2 · Il Bound Esponenziale sull'Errore di Training

Teorema — Convergenza Esponenziale di AdaBoost

L'errore di training dell'ensemble H dopo T round soddisfa:

$$\operatorname{err}_{\text{train}}(H) \leq \prod_{t=1}^{T} Z_t \leq \exp\left(-2\sum_{t=1}^{T}\gamma_t^2\right)$$

dove $Z_t = 2\sqrt{\varepsilon_t(1-\varepsilon_t)}$ è il fattore di normalizzazione dei pesi e $\gamma_t = 1/2 - \varepsilon_t > 0$ è l'**edge** del t-esimo weak learner sopra il caso. Se ogni weak learner ha edge almeno $\gamma > 0$, l'errore di training decresce come $\exp(-2T\gamma^2)$.

DERIVAZIONE — Bound Esponenziale

$\operatorname{err}_{\text{train}} = \frac{1}{n}\sum_i \mathbf{1}[H(x_i) \neq y_i] \leq \frac{1}{n}\sum_i e^{-y_i f(x_i)}$ dove $f = \sum_t \alpha_t h_t$, poiché $\mathbf{1}[yf \leq 0] \leq e^{-yf}$. La 0-1 loss è dominata dalla perdita esponenziale

I pesi di AdaBoost soddisfano $w_i^{(T+1)} = \frac{1}{n} e^{-y_i f(x_i)}$ (per induzione, partendo da $w_i^{(1)} = 1/n$ e moltiplicando per i fattori esponenziali). Quindi $\frac{1}{n}\sum_i e^{-y_i f(x_i)} = \sum_i w_i^{(T+1)}$. Struttura moltiplicativa dei pesi

Per la struttura ricorsiva: $\sum_i w_i^{(t+1)} = Z_t \sum_i w_i^{(t)}$ con $Z_t = 2\sqrt{\varepsilon_t(1-\varepsilon_t)}$ (per il minimizzatore α_t ottimale). Con $\sum_i w_i^{(1)} = 1$: $\sum_i w_i^{(T+1)} = \prod_t Z_t$. Telescopio moltiplicativo

Per la disuguaglianza AM-GM: $2\sqrt{\varepsilon(1-\varepsilon)} \leq e^{-2\gamma^2}$ con $\gamma = 1/2 - \varepsilon$, dalla disuguaglianza $\sqrt{ab} \leq (a+b)/2$ e dallo sviluppo in serie. Bound finale sull'errore

Gradient Boosting: Derivazione Funzionale e Loss Generalizzate

Boosting come gradient descent nello spazio delle funzioni — pseudo-residui, shrinkage e stochastic gradient boosting.

325.1 · Il Gradient Descent Funzionale di Friedman

Friedman (2001) ha fornito la chiave concettuale che unifica il boosting: minimizzare $\mathcal{L}(F) = \sum_i L(y_i, F(x_i))$ rispetto alla funzione F è formalmente un problema di gradient descent nello *spazio delle funzioni*. Se F fosse un vettore in $\mathbb{R}^n$, il gradiente di $\mathcal{L}$ avrebbe componenti $\partial L(y_i, F)/\partial F(x_i) = -r_i$ con r_i i **pseudo-residui**: il passo di gradient descent aggiorna $F \leftarrow F + \eta\,\mathbf{r}$. Il Gradient Boosting approssima questo passo addestrando un albero h_m a fare regressione sui pseudo-residui $\{(x_i, r_i)\}$, ottenendo così un'approssimazione della direzione di discesa, e poi aggiorna $F_{m+1} = F_m + \eta\, h_m$.

Loss $L(y, F)$	**Pseudo-residuo** $r_i = -\partial L/\partial F$	**Applicazione**
$(y-F)^2/2$	$y_i - F(x_i)$	Regressione (classico)
$\lvert y - F\rvert$	$\text{sign}(y_i - F(x_i))$	Regressione robusta
$\log(1+e^{-yF})$	$y_i - \sigma(F(x_i))$	Classificazione binaria
Pinball_τ	$\tau\,\mathbf{1}[y > F] - (1-\tau)\,\mathbf{1}[y \le F]$	Quantile regression
Huber_δ	Residuo troncato a $\pm\delta$	Robusta + efficiente

Il **shrinkage** $\eta \in (0, 1]$ è il principale meccanismo di regolarizzazione: aggiornamenti piccoli per ogni albero richiedono più round ma generalizzano meglio, poiché il modello accumula lentamente complessità e può essere fermato prima dell'overfitting. Lo **Stochastic Gradient Boosting** (Friedman 2002) aggiunge il campionamento senza rimpiazzo di una frazione $s < 1$ dei dati a ogni round (tipicamente $s = 0.5$): riduce la correlazione tra gli alberi successivi (analoga alla decorrelazione del bagging), introduce regolarizzazione implicita, e riduce il costo computazionale per albero di un fattore $1/s$.

XGBoost: Secondo Ordine, Regolarizzazione e Ottimizzazioni Hardware

L'ingegneria del Gradient Boosting industriale — Taylor expansion, gain ottimale per split, weighted sketch e cache-awareness.

326.1 · L'Obiettivo Regolarizzato al Secondo Ordine

XGBoost (Chen e Guestrin, KDD 2016) ha dominato le competizioni Kaggle per anni con una serie di innovazioni rispetto al Gradient Boosting classico. L'obiettivo al round m, approssimato al secondo ordine (espansione di Taylor di L intorno a $\hat{y}_i^{(m-1)}$), è:

$$\mathcal{L}^{(m)} \approx \sum_i \left[g_i f_m(x_i) + \frac{1}{2} h_i f_m^2(x_i) \right] + \Omega(f_m), \quad \Omega(f) = \gamma T + \frac{1}{2}\lambda \|w\|^2$$

con $g_i = \partial L(y_i, \hat{y}^{(m-1)})/\partial \hat{y}^{(m-1)}$ (gradiente), $h_i = \partial^2 L/\partial(\hat{y}^{(m-1)})^2$ (curvatura), T il numero di foglie, e w_j i valori delle foglie. Fissata la struttura dell'albero, il valore ottimale di ogni foglia j e il valore dell'obiettivo corrispondente hanno forma chiusa:

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda}, \qquad \mathcal{L}_j^* = -\frac{1}{2} \frac{(\sum_{i \in I_j} g_i)^2}{\sum_{i \in I_j} h_i + \lambda} + \gamma$$

Il **gain** dello split che divide la foglia j in j_L e j_R è $\mathcal{L}_{j_L}^* + \mathcal{L}_{j_R}^* - \mathcal{L}_j^* - \gamma$: si esegue lo split solo se il gain è positivo (altrimenti la complessità aggiuntiva non è giustificata dai dati). Le innovazioni ingegneristiche includono: l'**approximate greedy algorithm** che usa quantili pesati per proporre candidati di split efficientemente su feature continue; il supporto nativo ai **valori mancanti** con direzione di split separata appresa dai dati; la **cache-awareness** per massimizzare l'hit rate della CPU cache durante il building degli alberi.

SVM Primale: Max-Margin, Hard e Soft Margin

Il classificatore lineare ottimale — la geometria del margine, hinge loss e il tradeoff C tra regolarizzazione e fit.

327.1 · La Geometria del Massimo Margine

Le **Support Vector Machines (SVM)** (Cortes e Vapnik 1995, basate sulla teoria VC di Vapnik e Chervonenkis degli anni '70) sono fondate su un'idea geometricamente intuitiva: tra tutti gli iperpiani $\{\mathbf{x} : \mathbf{w}^T\mathbf{x} + b = 0\}$ che separano due classi nel training set, quello a **massimo margine** — la massima distanza geometrica minima dai punti di training al confine decisionale — generalizza meglio su dati non visti. Il margine geometrico di un iperpiano $(\mathbf{w}, b)$ per un

esempio $(\mathbf{x}_i, y_i)$ con $y_i \in \{-1, +1\}$ è $\gamma_i = y_i(\mathbf{w}^T\mathbf{x}_i + b)/\|\mathbf{w}\|$; la larghezza del margine (distanza tra le due classi) è $2/\|\mathbf{w}\|$. Il problema primale dell'**hard-margin SVM** è:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \quad \text{s.t.} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \quad \forall i$$

Per dati non linearmente separabili, il **soft-margin SVM** (Cortes e Vapnik 1995) introduce variabili slack $\xi_i \geq 0$ che permettono violazioni del margine:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i \quad \text{s.t.} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i,\ \xi_i \geq 0$$

Il parametro $C > 0$ controlla il tradeoff: per $C \to \infty$ si recupera il hard-margin; per $C \to 0$ il classificatore ignora i dati e massimizza solo il margine. La loss per ogni esempio è la **hinge loss** $\ell(y, f) = \max(0, 1 - yf)$ — lineare nelle violazioni del margine, zero per gli esempi correttamente classificati con margine sufficiente. A differenza della log-loss, la hinge loss è esatta zero su un sottoinsieme dei punti di training (quelli con margine ≥ 1), producendo la sparsità dei support vectors che è la caratteristica distintiva delle SVM.

SVM Duale: Derivazione KKT Completa e Support Vectors

Il duale di Lagrange del problema primale — condizioni KKT, sparsità della soluzione e l'algoritmo SMO.

328.1 · Dal Primale al Duale tramite KKT

Il problema primale dell'SVM soft-margin è un programma quadratico convesso con vincoli lineari — le condizioni di KKT sono necessarie e sufficienti per l'ottimalità globale. Il Lagrangiano è:

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i - \sum_i \alpha_i[y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i] - \sum_i \mu_i\xi_i$$

DERIVAZIONE DUALE SVM

$\nabla_{\mathbf{w}}\mathcal{L} = 0 \Rightarrow \mathbf{w}^* = \sum_i \alpha_i y_i \mathbf{x}_i$: i pesi ottimali sono combinazione lineare dei dati di training pesata dai moltiplicatori KKT. Condizione di stazionarietà rispetto a $\mathbf{w}$

$\partial\mathcal{L}/\partial b = 0 \Rightarrow \sum_i \alpha_i y_i = 0$: bilanciamento dei moltiplicatori pesati per le etichette. Condizione di stazionarietà rispetto a b

$\partial\mathcal{L}/\partial\xi_i = 0 \Rightarrow C - \alpha_i - \mu_i = 0 \Rightarrow 0 \leq \alpha_i \leq C$ (per $\mu_i \geq 0$). Box constraint sui moltiplicatori: soft-margin

Sostituendo nel Lagrangiano si ottiene il **duale di Wolfe**: $\max_{\boldsymbol{\alpha}} \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j$ soggetto a $0 \leq \alpha_i \leq C$, $\sum_i \alpha_i y_i = 0$. Dipende solo da prodotti interni — porta al kernel trick

Le **condizioni di complementarietà KKT** $\alpha_i(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i) = 0$ e $\mu_i\xi_i = 0$ caratterizzano la struttura sparsa della soluzione: se $\alpha_i = 0$, l'esempio non contribuisce a $\mathbf{w}$ (classificato correttamente con margine ≥ 1); se $0 < \alpha_i < C$, l'esempio è esattamente sul margine ($\xi_i = 0$) ed è un **support vector**; se $\alpha_i = C$, l'esempio viola il margine ($\xi_i > 0$). Il classificatore finale dipende solo dai support vectors: $f(\mathbf{x}) = \sum_{i:\alpha_i>0} \alpha_i y_i \mathbf{x}_i^T\mathbf{x} + b$. L'algoritmo **SMO (Sequential Minimal Optimization)** (Platt 1998) risolve il duale ottimizzando solo due variabili α_i, α_j alla volta in forma analitica chiusa, garantendo la convergenza senza richiedere solutori QP generali.

Kernel SVM: Feature Space Implicito e RBF

SVM non lineare tramite kernel — kernelizzazione del duale, RBF e string kernels, RKHS e SMO kernel.

329.1 · La Kernelizzazione del Duale

Il duale dell'SVM dipende dai dati solo attraverso i prodotti interni $\mathbf{x}_i^T\mathbf{x}_j$. Sostituendo ogni prodotto interno con un kernel $k(\mathbf{x}_i, \mathbf{x}_j) = \langle\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)\rangle_{\mathcal{H}}$, si ottiene un SVM non lineare nello spazio originale senza mai computare ϕ esplicitamente. Il classificatore diventa:

$$f(\mathbf{x}) = \text{sign}\left(\sum_{i \in SV} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b\right)$$

Per il kernel **RBF** $k(\mathbf{x}, \mathbf{x}') = \exp(-\gamma\|\mathbf{x} - \mathbf{x}'\|^2)$, la feature map ϕ è infinito-dimensionale (corrisponde a una serie di Taylor delle feature polinomiali pesate esponenzialmente) — il SVM trova un iperpiano separante in questo spazio implicito che corrisponde a un confine decisionale non lineare nello spazio originale. Il kernel **polinomiale** $(c+\mathbf{x}^T\mathbf{x}')^d$ è equivalente a usare tutte le feature polinomiali di grado $\leq d$; i **string kernels** (Leslie et al. 2002) per sequenze DNA o testo contano sottosequenze comuni e operano su alfabeti discreti senza una rappresentazione vettoriale esplicita. La complessità del training con kernel è $O(n^2)$ in memoria (matrice di Gram) e $O(n^3)$ con QP naive; SMO riduce il costo in pratica a $O(n^2)$ o meno per dataset grandi con molti support vectors.

329.2 · Random Fourier Features: Approssimazione Scalabile dei Kernel

Il costo $O(n^2)$ in memoria per la matrice di Gram del kernel SVM diventa proibitivo per $n > 10^5$. La tecnica delle **Random Fourier Features (RFF)** (Rahimi e Recht, NeurIPS 2007) offre un'alternativa scalabile: invece di computare il kernel esatto, si approssima il prodotto interno $k(\mathbf{x}, \mathbf{x}') \approx z(\mathbf{x})^T z(\mathbf{x}')$ tramite una mappa esplicita di dimensione ridotta $z : \mathbb{R}^d o \mathbb{R}^D$ con $D \ll n$. Il fondamento teorico è il **Teorema di Bochner**: ogni kernel positivo semi-definito stazionario $k(\mathbf{x}-\mathbf{x}')$ è la trasformata di Fourier di una misura di probabilità $p(\boldsymbol{\omega})$ non negativa, $k(\boldsymbol{\delta}) = \int e^{i\boldsymbol{\omega}^T\boldsymbol{\delta}}\, p(\boldsymbol{\omega})\, d\boldsymbol{\omega}$. Per il kernel RBF con $\sigma = 1$, questa misura è gaussiana: $p(\boldsymbol{\omega}) = \mathcal{N}(\mathbf{0}, I)$. Campionando D frequenze $\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_D$ i.i.d. da $p(\boldsymbol{\omega})$ e costruendo la mappa $z(\mathbf{x}) = \sqrt{2/D}(\cos(\boldsymbol{\omega}_1^T\mathbf{x} + b_1), \ldots, \cos(\boldsymbol{\omega}_D^T\mathbf{x} + b_D))$ con sfasamenti $b_j \sim \mathcal{U}[0, 2\pi]$, si ottiene uno stimatore non-biasato del kernel: $\mathbb{E}[z(\mathbf{x})^T z(\mathbf{x}')] = k(\mathbf{x}, \mathbf{x}')$. La varianza decresce come $O(1/D)$, e con $D = O(d \log d/\varepsilon^2)$ si approssima il kernel con errore uniforme al massimo ε su qualunque dataset bounded con alta probabilità (via la disuguaglianza di Hoeffding). Le RFF trasformano il problema kernelizzato in un problema lineare nello spazio $\mathbb{R}^D$: il costo di training scende da $O(n^2 D)$ a $O(nD)$, permettendo di addestrare SVM con kernel approssimato su milioni di punti in pochi minuti.

Support Vectors, Margin Bound e la Teoria della Generalizzazione

Perché il massimo margine generalizza — VC theory, Rademacher complexity e il bound indipendente dalla dimensione.

330.1 · Il Margin Bound: Generalizzazione Indipendente dalla Dimensione

La giustificazione teorica del principio del massimo margine è uno dei risultati più eleganti della statistica computazionale. Per classificatori lineari in $\mathbb{R}^d$, la VC-dimension è $d+1$ — un bound pessimistico che non cattura la struttura del problema. Il **margin bound** fornisce un bound molto più fine che dipende dal margine anziché dalla dimensione:

Teorema — Margin Bound per SVM (Bartlett e Shawe-Taylor 1999)

Con probabilità almeno $1-\delta$ su scelta del training set, l'errore di generalizzazione dell'SVM con margine γ soddisfa:

$$P(\text{errore}) \leq \frac{4R^2}{\gamma^2 n}\left(1+\sqrt{\log n}\right)+\sqrt{\frac{\log(1/\delta)}{2n}}$$

dove $R=\max_i \|\mathbf{x}_i\|$ è il raggio dei dati. Il bound dipende dal rapporto R/γ (reciproco del margine normalizzato), ma *non dalla dimensione* d dello spazio di input — proprietà cruciale che spiega perché l'SVM con kernel RBF funziona in spazi di feature infinito-dimensionali.

Il **leave-one-out bound** fornisce un bound empirico ancora più diretto: $\text{err}_{\text{LOO}}(H) \leq \mathbb{E}[\#SV]/n$, dove il numero atteso di support vectors è una stima dell'errore LOO. Questo bound spiega la regola pratica che SVM con pochi support vectors (pochi punti con $\alpha_i > 0$) hanno bassa varianza e buona generalizzazione — i support vectors sono letteralmente i punti che "sostengono" il confine decisionale, e la loro frazione nel dataset è la misura della complessità effettiva del modello. Nella pratica industriale, SVM con kernel RBF e C, γ ottimizzati su un validation set raggiungono la migliore generalizzazione per dataset tabellari di media dimensione (da qualche migliaio a qualche decina di migliaia di esempi), rimanendo competitivi con le reti neurali in molti benchmark benchmark pubblici.

SVM Multi-Class e Calibrazione di Platt

Strategie multi-classe per SVM binari — one-vs-rest, one-vs-one, DAGSVM e la trasformazione in probabilità.

331.1 · Estensione alla Classificazione Multi-Classe

L'SVM è nativamente un classificatore binario; per $K > 2$ classi, le tre strategie principali differiscono per costo e accuratezza. La strategia **one-vs-rest (OvR)** addestra K SVM binari, ciascuno separando la classe k da tutte le altre; la predizione sceglie la classe con il massimo punteggio di decisione $f_k(\mathbf{x}) = \mathbf{w}_k^T\mathbf{x} + b_k$. Il problema è lo sbilanciamento: ogni SVM vede n/K esempi positivi e $n(K-1)/K$ negativi. La strategia **one-vs-one (OvO)** addestra $K(K-1)/2$ SVM binari, uno per ogni coppia di classi, e predice per votazione a maggioranza tra i $K(K-1)/2$ classificatori. Ogni SVM vede solo $2n/K$ esempi (bilanciati), ma il numero di classificatori cresce quadraticamente in K. **DAGSVM** (Platt, Cristianini e Shawe-Taylor 2000) organizza le decisioni OvO in un grafo aciclico diretto gerarchico, riducendo il numero di decisioni necessarie a predizione time da $K(K-1)/2$ a $K-1$ — un'efficienza particolarmente importante per K grande (centinaia di classi). La **Calibrazione di Platt** (Platt 1999) converte i punteggi di decisione $f(\mathbf{x})$ dell'SVM in probabilità fitting una sigmoide $P(y=1 \mid \mathbf{x}) = 1/(1+e^{Af(\mathbf{x})+B})$ su un validation set separato, permettendo di usare l'SVM come componente probabilistico in sistemi di ensemble, threshold-optimization o decision-making con costi asimmetrici.

331.2 · Structural SVM: il Massimo Margine per Output Strutturati

L'estensione più ambiziosa del principio SVM è la **Structural SVM** (Taskar, Guestrin e Koller 2003; Tsochantaridis, Hofmann, Joachims e Altun 2005), che porta il massimo margine a problemi con output di struttura combinatoria: sequenze, alberi sintattici, grafi, segmentazioni. Anziché apprendere K classificatori binari indipendenti, la Structural SVM apprende un'unica funzione di scoring $F(\mathbf{x}, \mathbf{y}; \mathbf{w}) = \langle \mathbf{w}, \boldsymbol{\Phi}(\mathbf{x}, \mathbf{y}) \rangle$ dove $\boldsymbol{\Phi}(\mathbf{x}, \mathbf{y})$ è una *feature map congiunta* che cattura le interazioni tra l'input $\mathbf{x}$ e l'intera struttura dell'output $\mathbf{y}$. La predizione è $\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} F(\mathbf{x}, \mathbf{y}; \mathbf{w})$, risolta da algoritmi di inferenza specifici al dominio (Viterbi per sequenze, CKY per alberi, graph cuts per segmentazione). Il training identifica ad ogni passo la *most violated constraint* — la struttura $\mathbf{y}$ più simile alla corretta ma con score troppo alto — e aggiunge il vincolo di margine corrispondente. La Structural SVM è stata lo stato dell'arte per il parsing sintattico e la traduzione statistica prima dell'era neurale, e i suoi principi permeano ancora il design delle loss functions per output strutturati: la CTC loss per il riconoscimento vocale, la sequence-level loss per la traduzione, e il SEARN per il reinforcement learning supervisionato devono tutti ai suoi fondamenti il concetto di margine strutturato.

Random Forests: Bagging, Decorrelazione e Feature Importance

Ensemble di alberi con randomizzazione — variance reduction, OOB error e l'interpretabilità tramite permutation importance.

332.1 · Dal Bagging alla Random Forest

Il **Bagging** (Bootstrap AGGregatING, Breiman 1996) riduce la varianza addestrando T istanze dello stesso learner ad alta varianza su bootstrap del dataset (campionamento con rimpiazzo di n esempi da n, escludendo in attesa circa il 37% dei punti originali) e aggregandone le predizioni. Per learner indipendenti identicamente distribuiti con varianza σ^2 e covarianza zero, la varianza dell'ensemble è σ^2/T. Nella pratica, gli alberi addestrati sullo stesso dominio non sono indipendenti: tendono a selezionare le stesse feature dominanti nella radice, producendo alta correlazione e un guadagno di varianza inferiore al bound teorico. Le **Random Forests** (Breiman 2001) introducono un secondo livello di randomizzazione: ad ogni split, si seleziona randomicamente solo un sottoinsieme di $\sqrt{d}$ feature (per classificazione) o $d/3$ feature (per regressione) tra cui cercare il miglior split. Questa *feature subsampling* de-correla gli alberi, approssimando la varianza teorica dell'ensemble indipendente. Il parametro $\sqrt{d}$ è quasi-ottimale: sufficiente a de-correlare ma non così piccolo da degradare la qualità dei singoli alberi.

La **Permutation Feature Importance** (Breiman 2001) stima il contributo di ogni feature alla performance: per ogni feature j, si permutano i suoi valori nel dataset di test (rompendo la sua correlazione con y) e si misura la degradazione dell'accuratezza $\Delta_j = \text{acc}_{\text{base}} - \text{acc}_{\text{permuted},j}$. Una feature con Δ_j alto è causalmente rilevante; una con $\Delta_j \approx 0$ è ridondante o irrilevante. L'**Out-of-Bag (OOB) error** sfrutta il 37% di esempi esclusi da ogni bootstrap: ciascun albero predice sui propri esempi OOB (quelli non visti durante il suo training), e aggregando le predizioni OOB di tutti gli alberi si ottiene una stima non-biasata dell'errore di generalizzazione — equivalente a una leave-one-out cross-validation, computazionalmente gratuita poiché viene calcolata durante il training.

Online Convex Optimization: il Framework Avversariale

Il gioco iterativo learner-avversario — regret cumulativo, no-regret algorithms e conversione online-to-batch.

333.1 · Il Protocollo dell'Online Learning Avversariale

L'**Online Convex Optimization (OCO)** (Hazan 2016, monografia) fornisce il framework teorico fondamentale per l'apprendimento sequenziale senza assunzioni distribuzionali sui dati. A differenza della statistica classica che assume i dati i.i.d. da una distribuzione fissa, l'OCO permette che la sequenza delle loss sia scelta da un avversario adattivo — un modello estremamente pessimistico che copre anche scenari di distributional shift, mercati finanziari, sistemi di controllo adattivo e qualunque ambiente non-stazionario.

Il protocollo si ripete per T round: il learner sceglie $\mathbf{w}_t \in \mathcal{K}$ (insieme convesso chiuso, il dominio delle decisioni); l'avversario rivela $\ell_t : \mathcal{K} \to \mathbb{R}$ (funzione di loss convessa, eventualmente scelta in base a $\mathbf{w}_t$); il learner subisce la loss $\ell_t(\mathbf{w}_t)$. L'obiettivo è minimizzare il **regret cumulativo** rispetto alla migliore decisione fissa in retrospettiva:

$$\text{Regret}_T = \sum_{t=1}^{T} \ell_t(\mathbf{w}_t) - \min_{\mathbf{w}^* \in \mathcal{K}} \sum_{t=1}^{T} \ell_t(\mathbf{w}^*)$$

Un algoritmo è **no-regret** se $\text{Regret}_T / T \to 0$ per $T \to \infty$: la sua performance media per round converge a quella del miglior predittore fisso, anche se questo predittore è scelto con la conoscenza completa di tutte le loss future. La generalità del framework OCO è straordinaria: ogni algoritmo no-regret si converte automaticamente in un algoritmo di learning PAC tramite la *online-to-batch conversion* — si usa la media $\bar{\mathbf{w}}_T = \frac{1}{T}\sum_t \mathbf{w}_t$ come predittore batch, e il bound sul regret diventa un bound sul rischio empirico.

Follow The Regularized Leader e Online Gradient Descent

FTRL come algoritmo fondamentale — l'instabilità di FTL, la regolarizzazione come stabilizzatore e il regret $O(\sqrt{T})$.

334.1 · FTL e la Necessità della Regolarizzazione

La scelta greedy naturale nell'OCO è **Follow The Leader (FTL)**: ad ogni round, scegliere la decisione che minimizza le loss cumulative passate: $\mathbf{w}_{t+1} = \arg\min_{\mathbf{w} \in \mathcal{K}} \sum_{\tau=1}^{t} \ell_\tau(\mathbf{w})$. FTL funziona bene quando le loss sono "regolari" (fortemente convesse, bassa varianza), ma un avversario può costruire sequenze di loss lineari che forzano FTL a oscillare tra due estremi del

dominio, ottenendo regret lineare $\Omega(T)$. La soluzione è aggiungere un termine di **regolarizzazione** $R(\mathbf{w})$ che penalizza le decisioni lontane da un punto fisso, stabilizzando gli aggiornamenti:

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}\in\mathcal{K}} \left[\eta R(\mathbf{w}) + \sum_{\tau=1}^{t} \ell_\tau(\mathbf{w}) \right]$$

Per $R(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2$ (regolarizzazione L2) e loss lineari $\ell_t(\mathbf{w}) = \mathbf{g}_t^T\mathbf{w}$, FTRL si riduce a $\mathbf{w}_{t+1} = \Pi_{\mathcal{K}}(-\frac{1}{\eta}\sum_{\tau=1}^{t}\mathbf{g}_\tau)$ — che per il caso non proiettato è equivalente all'**Online Gradient Descent (OGD)**: $\mathbf{w}_{t+1} = \Pi_{\mathcal{K}}(\mathbf{w}_t - \eta\mathbf{g}_t)$. Il regret bound fondamentale di OGD (con $\eta = D/(G\sqrt{T})$, D = diametro di $\mathcal{K}$, G = max norma dei gradienti) è:

$$\text{Regret}_T \leq \frac{D^2}{2\eta} + \frac{\eta}{2}\sum_t \|\mathbf{g}_t\|^2 \leq DG\sqrt{T}$$

Il bound $O(\sqrt{T})$ è quasi-ottimale: esiste un lower bound $\Omega(\sqrt{T})$ per qualunque algoritmo no-regret su funzioni convesse Lipschitz — OGD è quindi minimax ottimale a meno di costanti. Per loss **fortemente convesse** (con modulo $\lambda > 0$: $\ell_t(\mathbf{w}) \geq \ell_t(\mathbf{v}) + \nabla\ell_t(\mathbf{v})^T(\mathbf{w}-\mathbf{v}) + \frac{\lambda}{2}\|\mathbf{w}-\mathbf{v}\|^2$) il bound migliora a $O(\frac{G^2}{\lambda}\log T)$ con learning rate $\eta_t = 1/(\lambda t)$ — un miglioramento esponenziale da $\sqrt{T}$ a $\log T$ che spiega l'efficienza aggiuntiva di OGD su funzioni regolari come la log-loss.

L'Algoritmo Hedge: Exponential Weights per la Prediction with Expert Advice

FTRL con entropia come regolarizzazione — EWA, il regret logaritmico in N e l'ottimalità del bound $\sqrt{T\log N}$.

335.1 · Il Setting dell'Expert Advice

La **prediction with expert advice** è un caso speciale dell'OCO con dominio il simplesso $\Delta_N = \{p \in \mathbb{R}^N : p_i \geq 0, \sum_i p_i = 1\}$ e la regolarizzazione naturale è l'**entropia negativa** $R(\mathbf{p}) = \sum_i p_i \log p_i$. A ogni round, si dispone di N "esperti" con predizioni/strategie distinte, si sceglie una distribuzione $p_t \in \Delta_N$ su di essi, si osservano le loss $\ell_t(i)$ di ogni esperto, e si subisce la loss attesa $\mathbf{p}_t^T\boldsymbol{\ell}_t$. FTRL con regolarizzazione entropica produce l'**algoritmo Hedge** (Freund e Schapire 1997, detto anche Exponential Weights Algorithm):

$$p_{t+1}(i) = \frac{p_t(i)\, e^{-\eta \ell_t(i)}}{\sum_j p_t(j)\, e^{-\eta \ell_t(j)}}$$

Partendo dall'uniforme $p_1(i) = 1/N$, i pesi si aggiornano moltiplicativamente: ogni esperto viene "scontato" esponenzialmente con la sua loss accumulata. Il bound sul regret dell'algoritmo Hedge con $\eta = \sqrt{8 \log N / T}$ è:

$$\text{Regret}_T \leq \sqrt{\frac{T \log N}{2}}$$

Il fatto notevole è la dipendenza da $\log N$ invece di N: il regret cresce *logaritmicamente* nel numero di esperti. Questo permette di competere con pool astronomicamente grandi di esperti: $N = 2^{100}$ strategie binarie richiedono solo $\sqrt{100T/2}$ di regret aggiuntivo rispetto alla migliore strategia singola. Questa proprietà è alla base dei teoremi di learning in classi di ipotesi di dimensione esponenziale e degli algoritmi di boosting (AdaBoost è Hedge applicato alle ipotesi del weak learner con una specifica sequenza di loss). L'aggiornamento moltiplicativo è computazionalmente efficiente ($O(N)$ per round) e numericamente stabile se implementato nel dominio logaritmico (log-sum-exp trick).

Regret Bounds: Ottimalità Minimax e AdaGrad Adattivo

Lower bounds per il regret — argomenti di minimax, bound adattivi e la superiorità di AdaGrad sui gradienti sparsi.

336.1 · Lower Bound Minimax per l'OCO

Il bound $O(\sqrt{T})$ di OGD è fondamentalmente ottimale: qualunque algoritmo deterministico o randomizzato deve pagare $\Omega(\sqrt{T})$ di regret su sequenze avversariali di loss convesse Lipschitz. La **prova del lower bound** usa l'argomento della "sequenza difficile": si considera un avversario che sceglie la loss lineare che massimizza il regret dato $\mathbf{w}_t$ (una variante del problema del partizionamento), e si dimostra che il regret accumulato è almeno $\Omega(DG\sqrt{T})$. Per OGD, il bound superiore $DG\sqrt{T}$ e il lower bound $\Omega(DG\sqrt{T})$ si incontrano, rendendolo minimax ottimale. La dipendenza da D e G è anch'essa ottimale: raddoppiare il diametro o la Lipschitz constant raddoppia il regret necessario.

Per gradienti **sparsi o eterogenei**, il bound $O(\sqrt{T})$ è eccessivamente pessimistico: se la maggior parte delle componenti del gradiente sono zero, OGD con learning rate globale η è inefficiente. Il regret bound di **AdaGrad** (Duchi, Hazan e Singer 2011) è adattivo:

$$\text{Regret}_T^{\text{AdaGrad}} \leq O\left(\sqrt{\sum_t \|\mathbf{g}_t\|^2}\right) \leq O(\sqrt{T})$$

ma con costante che dipende dalla somma cumulativa dei quadrati dei gradienti — non dal massimo. Se i gradienti sono sparsi (la maggior parte delle componenti zero in ogni round), $\sum_t \|\mathbf{g}_t\|^2 \ll TG^2$ e AdaGrad ottiene un regret molto inferiore a $DG\sqrt{T}$. Questa proprietà di adattività — implicita nella struttura della precondizionatura adattativa — è la ragione teorica della superiorità di AdaGrad (e Adam) rispetto a SGD con learning rate fisso per problemi di NLP con vocabolari grandi, dove la maggior parte dei parametri degli embedding ricevono gradienti rari.

Il Problema degli Esperti: Weighted Majority e Mistake Bounds

Apprendimento con feedback binario — l'algoritmo WM, le mistake bounds, e la differenza tra regret e mistake setting.

337.1 · Il Modello delle Mistake Bounds

Prima del framework OCO del regret, Littlestone e Warmuth (1994) avevano analizzato il **modello delle mistake bounds** per la predizione binaria con esperti: si vuole limitare il numero totale di errori M dell'algoritmo in funzione degli errori M^* del miglior esperto. L'**algoritmo Weighted Majority (WM)** mantiene pesi $w_i > 0$ per ogni esperto (inizializzati a 1), predice la classe di maggioranza pesata, e dopo ogni round moltiplica per $\beta \in (0,1)$ i pesi degli esperti che hanno sbagliato. Il bound sulle mistake è:

$$M \leq \frac{\log(1/w^*(T+1))}{\log(1/\beta)} = \frac{M^* \log(1/\beta) + \log N}{\log(1/\beta)} = M^* + \frac{\log N}{\log(1/\beta)}$$

Con $\beta = 1/2$ si ottiene $M \leq 2M^* + 2\log_2 N$ — il numero di errori dell'algoritmo è al più il doppio del miglior esperto più un termine logaritmico in N. Questo è un **absolute bound** (sul

numero totale di errori), più forte del regret bound relativo (su loss cumulative). La differenza fondamentale tra il modello WM e Hedge è che WM opera con loss $\{0,1\}$ e garantisce bounds assoluti sul numero di errori; Hedge opera con loss continue in $[0,1]$ e garantisce bounds sul regret relativo. Per problemi dove la classe corretta è deterministica e il miglior esperto fa zero errori (setting *realizable*), WM garantisce $M \leq 2\log_2 N$ errori indipendentemente dalla lunghezza della sequenza — una garanzia molto forte che non è coperta dal framework del regret.

337.2 · Tracking the Best Expert: Regret Adattivo e Cambio di Distribuzione

L'algoritmo Hedge minimizza il regret rispetto al *miglior esperto fisso* — la strategia statica ottimale sull'intera sequenza di T round. In molti scenari reali, tuttavia, il miglior esperto cambia nel tempo: un modello meteorologico può essere superiore in estate ma inferiore in inverno, una strategia di trading può funzionare in un regime di mercato ma non in un altro. Il **tracking regret** misura la performance rispetto alla migliore sequenza di esperti che cambia S volte: $\text{Regret}_T^{(S)} = \sum_t \ell_t(\mathbf{p}_t) - \min_{\pi:|\text{cambi}|\leq S} \sum_t \ell_t(\pi_t)$. Minimizzare il tracking regret richiede di rilevare i cambi di regime e "dimenticare" il passato in modo adattivo. L'algoritmo **Fixed Share** (Herbster e Warmuth 1998) modifica Hedge aggiungendo un termine di mescolamento uniforme con probabilità α: ad ogni round, con probabilità $1-\alpha$ si aggiorna normalmente, con probabilità α si redistribuisce il peso uniformemente tra tutti gli esperti. Il parametro α controlla il tradeoff: valori alti permettono adattamento rapido ma aumentano il regret statico; valori bassi viceversa. Il tracking regret di Fixed Share è $O(\sqrt{T(S\log N + S\log(T/S))})$ con $\alpha = S/T$ ottimale — un bound che cresce solo logaritmicamente in S quando $S \ll T$, riflettendo il fatto che il cambio di regime porta informazione progressiva sull'ambiente. Questa proprietà ha reso Fixed Share e le sue varianti (AdaHedge, Adaptive Follow the Regularized Leader) gli algoritmi di riferimento per l'online learning in ambienti non stazionari come i mercati finanziari e il routing adattivo nelle reti di telecomunicazione.

EXP3: Exploration-Exploitation con Feedback Parziale

Il bandit avversariale — l'importance sampling estimator, il gap informazionale e il regret $O(\sqrt{KT\log K})$.

338.1 · Il Setting Bandit: Feedback Parziale come Modello Fondamentale

Nel framework bandit, il learner può scegliere tra K azioni ma osserva *solo la loss dell'azione scelta* — non le loss delle azioni alternative. Questo modella situazioni reali dove l'informazione è intrinsecamente parziale: un motore di raccomandazione mostra un contenuto e osserva il click-through ma non cosa sarebbe successo con contenuti diversi; un algoritmo di pricing fissa un prezzo e osserva se il cliente acquista, ma non il prezzo massimo che avrebbe pagato; un medico prescrive una terapia e osserva l'esito del paziente, ma non gli esiti controfattuali delle terapie alternative. Il gap informazionale tra il setting *full-information* (dove si osservano le loss di tutte le azioni) e il setting *bandit* è fondamentale: il regret ottimale nel bandit avversariale è $\Theta(\sqrt{KT})$ invece di $\Theta(\sqrt{T \log K})$ — un fattore moltiplicativo $\sqrt{K/\log K}$ che riflette il costo della mancanza di informazione.

L'algoritmo **EXP3 (Exponential-weight algorithm for Exploration and Exploitation)** (Auer, Cesa-Bianchi, Freund e Schapire 2002) risolve il bandit avversariale con un meccanismo elegante. Si mantengono pesi $w_t(k)$ per ogni azione (inizializzati a 1), si campiona l'azione $I_t \sim p_t$ con $p_t(k) \propto w_t(k)$, si osserva la loss $\ell_t(I_t)$ dell'azione scelta, e si costruisce lo **stimatore di importance sampling** per la loss di ogni azione:

$$\tilde{\ell}_t(k) = \begin{cases} \ell_t(k)/p_t(k) & \text{se } I_t = k \\ 0 & \text{altrimenti} \end{cases}, \qquad \mathbb{E}[\tilde{\ell}_t(k)] = \ell_t(k)$$

Si aggiornano poi i pesi come in Hedge: $w_{t+1}(k) = w_t(k)\exp(-\eta\tilde{\ell}_t(k))$. Lo stimatore $1/p_t(k)$ è non-biasato: l'azione scelta raramente ($p_t(k)$ piccolo) riceve una stima di loss alta quando viene selezionata, compensando la rarità. La varianza dello stimatore è $O(1/p_t(k))$ — alta per le azioni selezionate raramente, bassa per quelle frequenti. Il regret di EXP3 con $\eta = \sqrt{2\log K/(KT)}$ è $\text{Regret}_T \leq O(\sqrt{KT\log K})$, quasi-ottimale rispetto al lower bound $\Omega(\sqrt{KT})$.

UCB Stocastico: Regret Logaritmico e il Lower Bound di Lai-Robbins

Il bandit multi-armed stocastico — UCB1, l'ottimalità rispetto al bound di Lai-Robbins e la struttura del problema.

339.1 · Banditi Stocastici e il Bound Logaritmico

Nel **bandit stocastico**, le reward di ogni azione k sono campionate i.i.d. dalla distribuzione $\mathcal{D}_k$ con media μ_k: l'ambiente non è avversariale ma stazionario. Il regret cumulativo è $\sum_{t=1}^{T}(\mu^* - \mu_{I_t})$ con $\mu^* = \max_k \mu_k$. Lai e Robbins (1985) hanno stabilito il primo **lower bound logaritmico**: qualunque algoritmo consistente (che ha regret $o(T^\alpha)$ per ogni $\alpha > 0$) deve selezionare ogni azione sub-ottimale k almeno $\Omega(\log T / \mathrm{KL}(\mathcal{D}_k \| \mathcal{D}_{k^*}))$ volte, producendo un regret cumulativo almeno $\Omega(\sum_{k:\mu_k<\mu^*} \frac{\Delta_k \log T}{\mathrm{KL}(\mathcal{D}_k \| \mathcal{D}_{k^*})})$ con $\Delta_k = \mu^* - \mu_k$. Questo lower bound logaritmico — in netto contrasto con il $\Theta(\sqrt{KT})$ avversariale — riflette la "struttura benigna" del setting stocastico: la stazionarietà della distribuzione permette all'agente di imparare le medie con crescente precisione, riducendo la necessità di exploration.

L'algoritmo **UCB1** (Auer, Cesa-Bianchi e Fischer 2002) è semplice e asintoticamente ottimale: a ogni round si seleziona l'azione con il massimo *Upper Confidence Bound*:

$$I_t = \arg\max_k \left[\hat{\mu}_k(t) + \sqrt{\frac{2 \log t}{n_k(t)}} \right]$$

dove $\hat{\mu}_k$ è la media campionaria e n_k il numero di selezioni dell'azione k fino al round t. Il termine $\sqrt{2 \log t / n_k}$ è derivato dalla disuguaglianza di Hoeffding e garantisce con probabilità $1 - 1/t^4$ che $\mu_k \leq \hat{\mu}_k + \sqrt{2 \log t / n_k}$. UCB1 bilancia sfruttamento (selezionare l'azione con la stima media più alta) ed esplorazione (selezionare azioni poco campionate, con UCB alto) in modo automatico, senza parametri da settare. Il regret totale è $O(\sum_{k:\Delta_k>0} \frac{\log T}{\Delta_k})$ — logaritmico in T e inversamente proporzionale ai gap Δ_k (le azioni con gap piccolo richiedono più esplorazione per essere identificate come sub-ottimali).

Contextual Bandits: LinUCB, Neural Bandits e Recommendation

Sfruttare il contesto per la selezione delle azioni — regressione lineare online, NeuralUCB e i sistemi di raccomandazione.

340.1 · Il Setting Contextual Bandit

I **contextual bandits** generalizzano i banditi stocastici introducendo un contesto $\mathbf{x}_t$ osservato prima di scegliere l'azione: il learner deve scegliere $a_t \in [K]$ dopo aver visto $\mathbf{x}_t$, con reward $r_t(a_t) = \mu_{a_t}(\mathbf{x}_t) + \varepsilon_t$. Il regret si misura rispetto alla policy ottimale contestuale

$\pi^*(\mathbf{x}) = \arg\max_a \mu_a(\mathbf{x})$. Il setting modella la raccomandazione di contenuti (contesto = utente + item features, azione = quale item mostrare), il pricing dinamico (contesto = caratteristiche del cliente e del prodotto, azione = prezzo), e la personalizzazione medica (contesto = caratteristiche del paziente, azione = terapia). A differenza del bandit non-contestuale, la policy ottimale dipende dal contesto e deve essere appresa in modo funzionale — non basta stimare una singola media per azione.

LinUCB (Li, Chu, Langford e Schapire, WWW 2010) assume $\mu_a(\mathbf{x}) = \mathbf{x}^T\boldsymbol{\theta}_a$ (modello lineare) e mantiene una stima ridge-regression per ogni azione: $\hat{\boldsymbol{\theta}}_a = (A_a)^{-1}\mathbf{b}_a$ con $A_a = I + \sum_{t:I_t=a}\mathbf{x}_t\mathbf{x}_t^T$ e $\mathbf{b}_a = \sum_{t:I_t=a} r_t\mathbf{x}_t$. L'UCB contestuale è $\mathbf{x}^T\hat{\boldsymbol{\theta}}_a + \alpha\sqrt{\mathbf{x}^T A_a^{-1}\mathbf{x}}$ dove il secondo termine è l'incertezza della predizione lineare nel punto $\mathbf{x}$. Il regret di LinUCB è $\tilde{O}(\sqrt{T \cdot d\log(1+Tn/d)})$ dove d è la dimensione del contesto. I **Neural Bandits** (NeuralUCB, Zhou, Gu e Gu 2020; NeuralTS) estendono questo schema usando reti neurali per modellare $\mu_a(\mathbf{x})$: l'incertezza è stimata tramite il gradiente della rete (Neural Tangent Kernel regime) o tramite MC Dropout, permettendo di catturare relazioni contesto-reward non lineari complesse. Sistemi di raccomandazione industriali (Google News, TikTok, Spotify) usano varianti di LinUCB e Neural Bandits per bilanciare l'exploitation delle preferenze note dell'utente con l'exploration di nuovi contenuti.

Optimal Transport: Wasserstein Distance e il Teorema di Brenier

Trasporto ottimale come distanza tra distribuzioni — formulazione duale di Kantorovich, $\mathrm{W}_1 e W_2, applicazionial ML$.

341.1 · Il Problema di Monge e la Relaxazione di Kantorovich

Il problema del **Optimal Transport (OT)**, formulato originalmente da Gaspard Monge nel 1781 per il trasporto di terra in costruzione (minimizzare il lavoro totale per spostare una massa da una distribuzione a un'altra), ha trovato applicazioni profonde nel machine learning moderno come misura di distanza tra distribuzioni, per la generazione di dati, per il domain adaptation e per la valutazione di modelli generativi. La formulazione moderna di Leonid Kantorovich (1942, che gli valse il Nobel per l'economia nel 1975) è più trattabile: trovare il **piano di trasporto ottimale** $\gamma^* \in \Pi(p,q)$ (distribuzione congiunta con marginali p e q) che minimizza il costo atteso di trasporto:

$$W_c(p,q) = \inf_{\gamma\in\Pi(p,q)} \mathbb{E}_{(x,y)\sim\gamma}[c(x,y)]$$

dove $c(x, y)$ è il costo di trasportare una unità di massa da x a y (tipicamente $\|x - y\|^p$ per la distanza di Wasserstein W_p). La distanza di **Wasserstein-1** ($p = 1$, Earth Mover's Distance) ha la formulazione duale di Kantorovich-Rubinstein: $W_1(p, q) = \sup_{\|f\|_L \leq 1} \mathbb{E}_p[f(x)] - \mathbb{E}_q[f(y)]$ dove il supremo è sulle funzioni 1-Lipschitz — la formula usata nelle WGAN per addestrare il discriminatore "critico". La distanza di **Wasserstein-2** ($p = 2$) ha la formulazione tramite il **Teorema di Brenier** (1991): il piano di trasporto ottimale è deterministico (dato da una mappa $T^* : \mathbb{R}^d \to \mathbb{R}^d$) ed è il gradiente di una funzione convessa $T^* = \nabla\psi^*$ — una connessione profonda con la convex analysis. Il calcolo esatto di W_p è un problema di linear programming con $O(n^2)$ variabili per distribuzioni discrete con n atomi; il **Sinkhorn algorithm** (Cuturi 2013) introduce una regolarizzazione entropica $\varepsilon \sum_{ij} \gamma_{ij} \log \gamma_{ij}$ che rende il problema strettamente convesso e risolvibile in $O(n^2/\varepsilon^2)$ iterazioni di normalizzazione alternata delle righe e colonne — permettendo il calcolo efficiente e differenziabile di OT per applicazioni di deep learning come il training di modelli generativi, l'allineamento di embedding multilingue e la costruzione di atlas anatomici.

---

ewpage

Notazione Matematica e Convenzioni

Riepilogo sistematico dei simboli e delle convenzioni usate nel Volume V.

A.1 · Vettori, Matrici e Tensori

I **vettori** sono denotati con lettere minuscole in grassetto: $\mathbf{x} \in \mathbb{R}^d$; le **matrici** con lettere maiuscole in grassetto: $\mathbf{W} \in \mathbb{R}^{m \times n}$; i **tensori** di ordine superiore con lettere calligrafiche: $\mathcal{T}$. La **trasposta** è ^{T}: $\mathbf{w}^T$ è un vettore riga. Il **prodotto scalare** è $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^T\mathbf{y} = \sum_i x_i y_i$. Il **prodotto di Kronecker** è $A \otimes B$ (matrice a blocchi con blocco (i, j) uguale a $a_{ij}B$). Il **prodotto di Hadamard** (element-wise) è $\mathbf{x} \odot \mathbf{y}$. Il **prodotto esterno** è $\mathbf{x}\mathbf{y}^T \in \mathbb{R}^{d \times d}$.

Le **norme vettoriali** principali sono: L^1: $\|\mathbf{x}\|_1 = \sum_i |x_i|$; euclidea L^2: $\|\mathbf{x}\|_2 = \sqrt{\sum_i x_i^2}$; L^∞: $\|\mathbf{x}\|_\infty = \max_i |x_i|$. Per le **matrici**: Frobenius $\|A\|_F = \sqrt{\mathrm{tr}(A^T A)}$; spettrale $\|A\|_2 = \sigma_{\max}(A)$; nucleare $\|A\|_* = \sum_i \sigma_i$. La **traccia** soddisfa $\mathrm{tr}(ABC) = \mathrm{tr}(CAB) = \mathrm{tr}(BCA)$ (ciclicità). Le identità di differenziazione matriciale usate nel testo: $\nabla_{\mathbf{x}}(\mathbf{a}^T\mathbf{x}) = \mathbf{a}$; $\nabla_{\mathbf{x}}(\mathbf{x}^T A\mathbf{x}) = (A + A^T)\mathbf{x}$; $\nabla_A \log\det A = A^{-T}$; $\nabla_A \mathrm{tr}(A^T B) = B$.

A.2 · Calcolo Differenziale

Il **gradiente** di $f : \mathbb{R}^d \to \mathbb{R}$ è il vettore colonna $\nabla_{\mathbf{x}} f = (\partial f/\partial x_1, \ldots, \partial f/\partial x_d)^T$. La **matrice jacobiana** di $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$ ha entrata (i,j) uguale a $\partial f_i/\partial x_j$. La **matrice hessiana** è $H_{ij} = \partial^2 f/\partial x_i \partial x_j$. Il **gradiente riemanniano** (§314) è la proiezione del gradiente euclideo sul tangent space: $\text{grad}_{\mathcal{M}} f = \mathcal{P}_{T_p\mathcal{M}}(\nabla f)$. L'**exponential map** $\exp_p : T_p\mathcal{M} \to \mathcal{M}$ mappa un vettore tangente al punto corrispondente sulla varietà; la **logarithmic map** $\log_p$ è la sua inversa locale.

A.3 · Probabilità e Divergenze

Le **distribuzioni comuni**: gaussiana multivariata $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$; Bernoulli $\text{Ber}(p)$; categorica $\text{Cat}(\boldsymbol{\pi})$; Dirichlet $\text{Dir}(\boldsymbol{\alpha})$; Gamma $\text{Gam}(a,b)$; Student-t t_ν. La **KL-divergenza** tra gaussiane ha forma chiusa: $\text{KL}(\mathcal{N}_1\|\mathcal{N}_2) = \frac{1}{2}[\text{tr}(\Sigma_2^{-1}\Sigma_1) + (\boldsymbol{\mu}_2-\boldsymbol{\mu}_1)^T\Sigma_2^{-1}(\boldsymbol{\mu}_2-\boldsymbol{\mu}_1) - d + \log|\Sigma_2|/|\Sigma_1|]$. La **distanza di Wasserstein-2** tra gaussiane è $W_2^2(\mathcal{N}_1,\mathcal{N}_2) = \|\boldsymbol{\mu}_1-\boldsymbol{\mu}_2\|^2 + \text{tr}[\Sigma_1+\Sigma_2-2(\Sigma_1^{1/2}\Sigma_2\Sigma_1^{1/2})^{1/2}]$. L'**ELBO** per un modello latente è $\mathcal{L}(q) = \mathbb{E}_q[\log P(\mathbf{x},\mathbf{z}) - \log q(\mathbf{z})] = \log P(\mathbf{x}) - \text{KL}(q\|P(\cdot|\mathbf{x}))$.

Strumenti Matematici Essenziali

Teoremi e disuguaglianze usati nelle dimostrazioni del Volume V.

B.1 · Disuguaglianze Fondamentali

La **disuguaglianza di Jensen**: per φ convessa, $\varphi(\mathbb{E}[X]) \le \mathbb{E}[\varphi(X)]$. Applicazione diretta: poiché $-\log$ è convessa, si ha $\log \mathbb{E}_q[P/q] \ge \mathbb{E}_q[\log(P/q)]$, da cui l'ELBO è un lower bound al log-evidence. La **disuguaglianza di Hoeffding** per variabili bounded $X_i \in [a_i, b_i]$ i.i.d.: $P(\bar{X} - \mathbb{E}[\bar{X}] \ge t) \le \exp(-2n^2t^2/\sum_i (b_i-a_i)^2)$ — base dei confidence bounds in UCB. La **disuguaglianza di Pinsker**: $\|P-Q\|_{TV}^2 \le \frac{1}{2}\text{KL}(P\|Q)$ — lega variazione totale e KL. Il **lemma di Azuma** per martingale con differenze bounded $|X_t - X_{t-1}| \le c_t$: $P(X_T - X_0 \ge t) \le \exp(-2t^2/\sum_\tau c_\tau^2)$ — generalizzazione di Hoeffding per sequenze dipendenti, alla base dei regret bounds online.

La **disuguaglianza AM-GM**: $\frac{1}{n}\sum_i x_i \ge (\prod_i x_i)^{1/n}$ per $x_i \ge 0$ — usata nella prova del bound esponenziale di AdaBoost. La **disuguaglianza di Cauchy-Schwarz**: $|\langle f, g\rangle|^2 \le \langle f, f\rangle\langle g, g\rangle$ — usata nel margine SVM e nelle norme RKHS. Il **Cramér-Rao bound**: $\text{Var}[\hat{\theta}] \ge F(\theta)^{-1}$ per qualunque stimatore non distorto — il lower bound fondamentale della statistica parametrica, con la Fisher information come suo inverso.

B.2 · Analisi Convessa e Dualità

Una funzione f è **convessa** se $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y)$; è μ-**fortemente convessa** se $f(y) \geq f(x) + \nabla f(x)^T(y-x) + \frac{\mu}{2}\|y-x\|^2$. La forte convessità garantisce unicità del minimizzatore e convergenza lineare del gradient descent. La **coniugata di Legendre-Fenchel**: $f^*(\mathbf{y}) = \sup_{\mathbf{x}}[\mathbf{y}^T\mathbf{x} - f(\mathbf{x})]$, sempre convessa. Per famiglie esponenziali, la log-partition $A(\boldsymbol{\eta})$ e la negativa dell'entropia $A^*(\boldsymbol{\mu})$ sono coniugate di Legendre l'una dell'altra — la struttura duale fondamentale della §288. La **dualità forte di Slater**: per programmi convessi con vincoli lineari, il gap duale è zero se esiste un punto strettamente ammissibile — giustifica la derivazione duale SVM in §328.

B.3 · Teoremi Spettrali e Approssimazioni

Il **Teorema Spettrale** per matrici simmetriche reali: ogni $A = A^T \in \mathbb{R}^{n\times n}$ è diagonalizzabile ortogonalmente $A = Q\Lambda Q^T$ con autovalori reali e autovettori ortonormali. Il **Teorema di Mercer**: ogni kernel PSD continuo si decompone come $k(x,x') = \sum_i \lambda_i \phi_i(x)\phi_i(x')$ con $\lambda_i \geq 0$ — la base teorica del RKHS e del kernel trick (§318). Il **Teorema di Approximazione Universale** (Cybenko 1989, Hornik 1991): reti con un singolo strato nascosto e funzione di attivazione sigmoide possono approssimare uniformemente qualunque funzione continua su un compatto. Il **Teorema di Brenier** (§341): il piano di trasporto ottimale per il costo $\|x-y\|^2$ è deterministico e dato dal gradiente di una funzione convessa $T^* = \nabla\psi^*$ — la connessione profonda tra OT e analisi convessa.

Pseudocodice degli Algoritmi Principali

Riepilogo algoritmico compatto per i metodi fondamentali del Volume V.

Algoritmo: Algoritmo 1 — Sum-Product su Factor Graph (Albero)

// Input: factor graph G=(V,F,E) ad albero, evidenze

Inizializza tutti i messaggi a 1 (o uniformi)

Forward (foglie $\to$ *radice*, *ordinetopologico*) :

$\mu_{x\to f(x)}$=$\prod_{b\in\partial x}\mu_{f_b\to x(x)}$

$\mu_{f \to x(x) = \Sigma_{x_{\partial f}} f(x_{\partial f}) \cdot \prod_{j \in \partial f} \mu_{x_j \to f(x_j)}}$

Backward (radice $\to$ $foglie$) : $simmetricoal forward$

Marginals: $\mathrm{P(x}_i) \propto \prod_{a \in \partial x_i} \mu_{f_a \to x_i(x_i)}$

Algoritmo: Algoritmo 2 — NUTS (No-U-Turn Sampler)

// Input: $U(\theta) = -logposterior, stepsize\varepsilon(adattivo)$

per t *= 1,. . . , T:*

p $N(0, M)$ *// campiona momento ausiliario*

$tree = BuildTree(\theta_t, p, \varepsilon, depth = 0)$

$\theta_{t+1} \leftarrow campioneuniformementedall'albero$

$BuildTree(\theta, p, \varepsilon, depth)$:

se *depth = 0: un passo leapfrog* $\to (\theta', p')$

altrimenti*: raddoppia in avanti o indietro*

ferma *se* $(\theta_+ - \theta_-) \cdot p_+ < 0 o (\theta_+ - \theta_-) \cdot p_- < 0$ *// U-turn*

Algoritmo: Algoritmo 3 — CAVI (Coordinate Ascent Variational Inference)

// Minimizza KL(q||P) massimizza ELBO

Inizializza*:* q_j $priorperj = 1, \ldots, m$

ripeti *fino a convergenza ELBO:*

per $j = 1,\ldots,m$:

$log\ q_j * (z_j) = E_{q_{-j}}[logP(x,z)] + const$

$normalizza \rightarrow q_j$

$ELBO = E_q[logP(x,z)] - E_q[logq(z)]$

ritorna $q^*(z) = \prod_j q_j(z_j)$

Algoritmo: Algoritmo 4 — AdaBoost

// *Input: n esempi* $(x_i, y_i), Tround$

Inizializza: $w_i = 1/n$

per $t = 1,\ldots,T$:

$h_t \leftarrow WeakLearneraddestratoconpesiw$

$\varepsilon_t = \Sigma_i w_i \cdot 1[h_t(x_i) \neq y_i]$

$\alpha_t = log((1-\varepsilon_t)/\varepsilon_t)$

$w_i \leftarrow w_i \cdot exp(-y_i \cdot \alpha_t \cdot h_t(x_i)); normalizza$

ritorna $H(x) = sign(\Sigma_t \alpha_t h_t(x))$

Algoritmo: Algoritmo 5 — EXP3 (Bandit Avversariale)

// *K azioni,* $\eta = \sqrt{}(2\ log\ K\ /\ KT)$

***Inizializza**: w(k) = 1 per k=1,...,K*

***per** t = 1,...,T:*

$p_t(k) = w_t(k)/\Sigma_j w_t(j)$

Campiona I_t p_t; osserva $\ell_t(I_t)$

$\ell_t(k) = \ell_t(k)/p_t(k) \cdot 1[I_t = k]$ *// importance sampling*

$w_{t+1(k)=w_t(k)\cdot exp(-\eta\cdot\ell_t(k))}$

Algoritmo: Algoritmo 6 — UCB1 (Bandit Stocastico)

// K azioni, reward bounded in [0,1]

***Inizializza**: seleziona ogni azione una volta; aggiorna μ_k, n_k*

***per** t = K+1,...,T:*

$I_t = argmax_k[\mu_k + \sqrt{}(2\ log\ t\ /\ n_k)]$

Osserva r_t; aggiorna $\mu_{I_t en_{I_t}}$

Complessità Computazionale dei Principali Algoritmi

Riepilogo dei costi di training, inferenza e memoria.

Metodo	*Training*	*Inferenza*	*Memoria*
GP esatto	$O(n^3)$	$O(n^2)$ *per punto*	$O(n^2)$
Sparse GP (FITC)	$O(nm^2)$	$O(m^2)$	$O(nm)$
SVGP (mini-batch b)	$O(bm^2 + m^3)$/*iter*	$O(m^2)$	$O(m^2)$
SVM (SMO)	$O(n^2)$–$O(n^3)$	$O(\lvert SV\rvert \cdot d)$	$O(n^2)$ *(Gram)*
AdaBoost (T round)	$O(T \cdot n \log n)$	$O(T)$	$O(T)$
Gradient Boosting	$O(T \cdot n \cdot d)$	$O(T \cdot D)$	$O(T \cdot 2^D)$
XGBoost	$O(T \cdot n \cdot d)$ *(sketch)*	$O(T \cdot D)$	$O(n \cdot d)$
HMC / NUTS	—	$O(L \cdot \nabla U)$/*camp.*	$O(d)$ *(adjoint)*
SGLD	—	$O(\tilde{\nabla} U)$/*passo*	$O(d)$
Mean Field VI	$O(T_{\text{VI}} \cdot n)$	$O(d_z)$	$O(d_z)$
Neural ODE (adjoint)	$O(L \cdot d)$ *(variabile)*	$O(L \cdot d)$	$O(d)$ *costante*
OGD (online)	$O(d)$/*round*	$O(d)$	$O(d)$
Hedge (online)	$O(N)$/*round*	$O(N)$	$O(N)$
UCB1 (bandit)	—	$O(K)$/*round*	$O(K)$
LinUCB (contextual)	$O(d^2)$/*round*	$O(Kd^2)$/*round*	$O(Kd^2)$
Sinkhorn OT	$O(n^2/\varepsilon^2)$	$O(n^2)$	$O(n^2)$

---

ewpage

Riferimenti Bibliografici

Opere fondative, articoli chiave e monografie di riferimento per le sezioni del Volume V.

Sezione A — Modelli Grafici Probabilistici

Pearl, J. *(1988). Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann. — Il testo fondativo delle Bayesian Networks e del belief propagation.*

Hammersley, J.M. & Clifford, P. *(1971). Markov Fields on Finite Graphs and Lattices. Preprint non pubblicato, citato universalmente. — La dimostrazione del teorema di fattorizzazione per MRF.*

Kschischang, F.R., Frey, B.J. & Loeliger, H.-A. *(2001). Factor graphs and the sum-product algorithm. IEEE Transactions on Information Theory, 47(2), 498–519.*

Yedidia, J.S., Freeman, W.T. & Weiss, Y. *(2001). Generalized belief propagation.*

NeurIPS 2001, 689–695. — Connessione tra loopy BP e Bethe Free Energy.

Wainwright, M.J. & Jordan, M.I. *(2008). Graphical models, exponential families, and variational inference. Foundations and Trends in Machine Learning, 1(1–2), 1–305. — Monografia di riferimento.*

Blei, D.M., Kucukelbir, A. & McAuliffe, J.D. *(2017). Variational inference: A review for statisticians. Journal of the American Statistical Association, 112(518), 859–877.*

Minka, T.P. *(2001). A Family of Algorithms for Approximate Bayesian Inference. PhD Thesis, MIT. — Expectation Propagation.*

Ranganath, R., Gerrish, S. & Blei, D.M. *(2014). Black box variational inference. AISTATS 2014.*

Kingma, D.P. & Welling, M. *(2013). Auto-encoding variational Bayes. ICLR 2014. — VAE con reparameterization trick.*

Hoffman, M.D. & Gelman, A. *(2014). The No-U-Turn Sampler. JMLR, 15, 1593–1623.*

Welling, M. & Teh, Y.W. *(2011). Bayesian learning via stochastic gradient Langevin dynamics. ICML 2011.*

Sezione B — Geometria dell'Informazione

Amari, S. *(1985). Differential-Geometrical Methods in Statistics. Springer.*

Amari, S. *(2016). Information Geometry and Its Applications. Springer.*

Amari, S. *(1998). Natural gradient works efficiently in learning. Neural Computation, 10(2), 251–276.*

Chentsov, N.N. *(1982). Statistical Decision Rules and Optimal Inference. AMS. — Unicità della metrica di Fisher.*

Martens, J. & Grosse, R. *(2015). Optimizing neural networks with Kronecker-factored approximate curvature. ICML 2015. — K-FAC.*

Duchi, J., Hazan, E. & Singer, Y. *(2011). Adaptive subgradient methods. JMLR, 12, 2121–2159. — AdaGrad.*

Kingma, D.P. & Ba, J. *(2015). Adam: A method for stochastic optimization. ICLR 2015.*

Sezione C — Neural ODE e Fisica

Chen, R.T.Q., Rubanova, Y., Bettencourt, J. & Duvenaud, D. *(2018). Neural ordinary differential equations. NeurIPS 2018.*

Grathwohl, W. et al. *(2019). FFJORD: Free-form continuous dynamics for scalable reversible generative models. ICLR 2019.*

Greydanus, S., Dzamba, M. & Yosinski, J. *(2019). Hamiltonian neural networks. NeurIPS 2019.*

Cranmer, M., Greydanus, S. & Ho, S. *(2020). Lagrangian neural networks. ICLR Workshop 2020.*

Raissi, M., Perdikaris, P. & Karniadakis, G.E. *(2019). Physics-informed neural networks. Journal of Computational Physics, 378, 686–707.*

Lu, L. et al. *(2021). Learning nonlinear operators via DeepONet. Nature Machine Intelligence, 3, 218–229.*

Lipman, Y. et al. *(2023). Flow matching for generative modeling. ICLR 2023.*

Sezione D — Inferenza Causale

Pearl, J. *(2000, 2009). Causality: Models, Reasoning and Inference. Cambridge University Press.*

Pearl, J. *(1995). Causal diagrams for empirical research. Biometrika, 82(4), 669–688. — Do-calculus originale.*

Shpitser, I. & Pearl, J. *(2006). Identification of joint interventional distributions in recursive semi-Markovian causal models. AAAI 2006. — Completezza del do-calculus.*

Pearl, J. & Mackenzie, D. *(2018). The Book of Why. Basic Books. — Gerarchia PCH.*

Spirtes, P., Glymour, C. & Scheines, R. *(2000). Causation, Prediction, and Search. MIT Press. — PC e FCI.*

Maathuis, M.H., Kalisch, M. & Bühlmann, P. *(2009). Estimating high-dimensional intervention effects. Annals of Statistics, 37(6A), 3133–3164. — IDA.*

Kusner, M.J. et al. *(2017). Counterfactual fairness. NeurIPS 2017.*

Bareinboim, E. & Pearl, J. *(2016). Causal inference and the data-fusion problem. PNAS,*

113(27), 7345–7352. — Transportability.

Sezione E — Geometria Profonda

do Carmo, M.P. *(1992). Riemannian Geometry. Birkhäuser. — Testo classico di geometria differenziale.*

Nickel, M. & Kiela, D. *(2017). Poincaré embeddings for learning hierarchical representations. NeurIPS 2017.*

Ganea, O., Bécigneul, G. & Hofmann, T. *(2018). Hyperbolic neural networks. NeurIPS 2018.*

Cohen, T.S. et al. *(2018). Spherical CNNs. ICLR 2018.*

Geiger, M. & Smidt, T. *(2022). e3nn: Euclidean neural networks. arXiv:2207.09453.*

Meyer, M. et al. *(2003). Discrete differential-geometry operators for triangulated 2-manifolds. In Visualization and Mathematics III. Springer. — Cotangent weights.*

Sharp, N. et al. *(2022). DiffusionNet: Discretization agnostic learning on surfaces. ACM SIGGRAPH 2022.*

Ungar, A.A. *(2008). Analytic Hyperbolic Geometry and Albert Einstein's Special Theory of Relativity. World Scientific. — Gyrovector spaces.*

Sezione F — Processi Gaussiani e Bayesian Optimization

Rasmussen, C.E. & Williams, C.K.I. *(2006). Gaussian Processes for Machine Learning. MIT Press. — Testo di riferimento, disponibile online.*

Snelson, E. & Ghahramani, Z. *(2006). Sparse Gaussian processes using pseudo-inputs. NeurIPS 2006. — FITC.*

Hensman, J., Matthews, A. & Ghahramani, Z. *(2015). Scalable variational Gaussian process classification. AISTATS 2015. — SVGP.*

Srinivas, N. et al. *(2012). Information-theoretic regret bounds for Gaussian process optimization. IEEE Trans. Inf. Theory, 58(5). — Regret GP-UCB.*

Russo, D. & Van Roy, B. *(2014). Learning to optimize via posterior sampling. Math. of Op. Research, 39(4). — Thompson Sampling.*

Liu, H., Simonyan, K. & Yang, Y. *(2019). DARTS: Differentiable architecture search.*

ICLR 2019.

Sezione G — Boosting e SVM

Freund, Y. & Schapire, R.E. *(1997). A decision-theoretic generalization of on-line learning and an application to boosting. JCSS, 55(1), 119–139. — AdaBoost, Premio Gödel 2003.*

Friedman, J.H. *(2001). Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29(5). — Gradient Boosting.*

Chen, T. & Guestrin, C. *(2016). XGBoost: A scalable tree boosting system. KDD 2016.*

Cortes, C. & Vapnik, V. *(1995). Support-vector networks. Machine Learning, 20(3), 273–297. — SVM originale.*

Platt, J. *(1998). Sequential minimal optimization. MSR-TR-98-14, Microsoft Research. — SMO.*

Vapnik, V. *(1998). Statistical Learning Theory. Wiley. — Teoria VC e margin bound.*

Breiman, L. *(2001). Random forests. Machine Learning, 45(1), 5–32.*

Sezione H — Online Learning e Regret

Hazan, E. *(2016). Introduction to Online Convex Optimization. Foundations and Trends in Optimization. — Monografia di riferimento, disponibile online.*

Cesa-Bianchi, N. & Lugosi, G. *(2006). Prediction, Learning, and Games. Cambridge University Press. — Testo fondamentale per l'online learning con esperti.*

Littlestone, N. & Warmuth, M.K. *(1994). The weighted majority algorithm. Information and Computation, 108(2), 212–261.*

Auer, P., Cesa-Bianchi, N., Freund, Y. & Schapire, R.E. *(2002). The nonstochastic multiarmed bandit problem. SIAM J. Comput., 32(1), 48–77. — EXP3.*

Auer, P., Cesa-Bianchi, N. & Fischer, P. *(2002). Finite-time analysis of the multiarmed bandit problem. Machine Learning, 47(2), 235–256. — UCB1.*

Lai, T.L. & Robbins, H. *(1985). Asymptotically efficient adaptive allocation rules. Adv. Appl. Math., 6(1), 4–22. — Lower bound logaritmico fondamentale.*

Li, L. et al. *(2010). A contextual-bandit approach to personalized news article recommendation. WWW 2010. — LinUCB.*

Cuturi, M. *(2013). Sinkhorn distances: Lightspeed computation of optimal transport distances. NeurIPS 2013.*

Villani, C. *(2008). Optimal Transport: Old and New. Springer. — La monografia completa.*